Cryptographic competitions

Daniel J. Bernstein[1,2]

[1] Department of Computer Science, University of Illinois at Chicago,
Chicago, IL 60607–7045, USA
[2] Horst Görtz Institute for IT Security, Ruhr University Bochum, Germany
`djb@cr.yp.to`

Abstract. Competitions are widely viewed as the safest way to select cryptographic algorithms. This paper surveys procedures that have been used in cryptographic competitions, and analyzes the extent to which those procedures reduce security risks.

Keywords: cryptography, competitions, DES, AES, eSTREAM, SHA-3, CAESAR, NISTPQC, NISTLWC

1 Introduction

> *The CoV individual reports point out several shortcomings and procedural weaknesses that led to the inclusion of the Dual EC DRBG algorithm in SP 800-90 and propose several steps to remedy them. . . . The VCAT strongly encourages standard development through open competitions, where appropriate.* —"NIST Cryptographic Standards and Guidelines Development Process: Report and Recommendations of the Visiting Committee on Advanced Technology of the National Institute of Standards and Technology" [**118**], 2014

Cryptographic competitions are not a panacea. DES, the output of the first cryptographic competition, had an exploitable key size (see [**50**], [**65**], [**125**], [**32**], and [**56**]), had an exploitable block size (see [**83**] and [**31**]), and at the same time had enough denials of exploitability (see, e.g., [**66**], [**49**, Section 7], [**68**], and [**1**]) to delay the deployment of stronger ciphers for decades. As another example, AES performance on many platforms relies on table lookups with secret indices ("S-table" or "T-table" lookups), and these table lookups were claimed to be "not vulnerable to timing attacks" (see [**48**, Section 3.3] and [**89**, Section 3.6.2]), but this claim was incorrect (see [**16**] and [**115**]), and this failure continues to cause security problems today (see, e.g., [**42**]). As a third example, SHA-3 was forced

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy—EXC 2092 CASA—390781972 "Cyber Security in the Age of Large-Scale Adversaries", by the U.S. National Science Foundation under grant 1913167, and by the Cisco University Research Program. "Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation" (or other funding agencies). Permanent ID of this document: `b7af715576cc229aaf8c532ea89bb6ace1c91a65`. Date: 2021.12.26.

to aim for a useless 2^{512} level of preimage security, and as a result is considerably larger and slower than necessary, producing performance complaints and slowing down deployment (see, e.g., [78])—which is a security failure if it means that applications instead use something weak (see, e.g., [81]) or nothing at all.

I don't mean to suggest that competitions are a bad idea. I can think of many more failures of cryptographic algorithms selected *without* competitions. But it is surprisingly difficult to find literature systematically analyzing the security risks in various algorithm-selection processes, and systematically working on designing processes that reduce these risks. Even if competitions are the best approach, there are many different types of competitions, and we should understand how these variations can avoid—or create—security risks.

This paper surveys, from the perspective of a skeptical security reviewer, the procedures used in cryptographic competitions. Given my role in organizing CAESAR, I have used CAESAR as a running example in the paper—among other things, reporting how CAESAR started, how it was run, and what it produced—but I have also used other cryptographic competitions as examples, including the DES, AES, eSTREAM, SHA-3, and NISTLWC competitions for symmetric algorithms, and the NISTPQC competition for asymmetric (public-key) algorithms. I hope that the analysis here is of value for future competitions.

1.1. Related work. For per-competition reports (at various levels of detail) on earlier competitions, see [49] for DES; [90] and [89] for AES; [100] for eSTREAM; and [98], [117], and [40] for SHA-3. NISTPQC and NISTLWC are ongoing but one can find preliminary reports at [8] and [9] for NISTPQC and [116] for NISTLWC. Beyond these reports, there is a vast literature on the security and performance of various submissions to the competitions.

The *procedures* used in cryptographic competitions are frequently mentioned as introductory material, but not as a core topic of cryptographic risk analysis. NIST's description of its cryptographic standardization process [47] includes a few paragraphs summarizing competition processes, saying that "competitions can focus the attention of cryptographers around the world". A deeper discussion of security-evaluation and performance-evaluation procedures—providing many reasons to wonder how effective competitions really are at reducing security risks—had appeared in [96, Sections 2 and 4] as part of input to the AES process from NESSIE, a joint project by European cryptography experts.

Security failures in cryptographic algorithms are traditionally blamed upon the specific algorithms that failed, and not upon the processes that selected those algorithms: cryptographers recommend against the algorithms while continuing to use the same processes. However, there has been some attention to the idea of protecting standardization processes against sabotage: see [118], [25], [30], and [54]. A standardization process can fail even when it is not under attack; in [23, Appendix B] I called for a systematic study of how reliably a standardization process produces secure standards. The idea that different processes can create different levels of risks was already a prerequisite for the common belief that competitions are less risky than typical standardization processes. The same idea is standard in the broader literature on risk analysis: see, e.g., [93].

2 Speed

A competition is defined by the Collins English Dictionary as “an event in which many people take part in order to find out who is best at a particular activity”.

One of the traditional forms of competition is a speed competition—a race. Who can swim the fastest? Who can run the fastest? Who can write a sorting program that runs the fastest? Who can drive the fastest? Who can build the fastest car? Who can write the fastest encryption program? Note that one can suppress the role of the humans in some of these questions, thinking of the cars as the competitors in a car race and thinking of the programs as the competitors in a software competition.

Sometimes speed competitions have a clear utilitarian purpose. Ancient Greek legend says that a runner ran 40 kilometers to Athens to report success in the Battle of Marathon. In selecting a runner for such a task, the commander wants to know, first, who can complete the task at all, and, second, who can complete it in the required amount of time. It would not be surprising if the original Marathon runner was selected as someone who did well enough in a previous race. Note that managers who have no idea what the required amount of time is will want to take specifically the *winner* of the competition, so as to reduce the risk of being too slow; someone who knows more about the requirements will tend to think that taking the winner is less important.

Sometimes people participate in or watch speed competitions simply for the thrill. People participating in a marathon today can reasonably claim health benefits, but people *watching* a marathon on the living-room television don’t have this excuse. Nobody claims that we need to know the fastest runner so as to decide who should be carrying a message to Athens.

If I set a speed record for some computation, am I doing it just for the thrill? Does the speed record actually matter for users? Making software run faster is a large part of my research; I want to *believe* that this is important, and I have an incentive to exaggerate its importance. People who select my software for its performance have a similar incentive to exaggerate the importance of this performance as justification for their decisions. Perhaps there is clear evidence that the performance is important, or perhaps not.

2.1. The machinery of cryptographic performance advertising, part 1: measurements. In 2019, a Google blog post [46] (see also the accompanying paper [45]) presented the following convincing story:

- Android had required storage encryption since 2015 *except* on “devices with poor AES performance (50 MiB/s and below)”.
- For example, on an ARM Cortex-A7, storage encryption with AES is “so slow that it would result in a poor user experience; apps would take much longer to launch, and the device would generally feel much slower”. (According to [12], the Cortex-A7 “has powered more than a billion smartphones”.)
- Starting in 2019, Android Pie was enabling storage encryption for these devices using Adiantum, a wide-block cipher built on top of ChaCha12.

- On a 1.19GHz Cortex-A7, AES-256-XTS decrypts at 20 MB/s. Adiantum decrypts at 112 MB/s.

What follows is another example of one cryptographic system solving the performance problems that caused another cryptographic system to be rejected.

Your Internet service provider used the Domain Name System (DNS) in three steps to find the address of `www.google.com`. It learned the address of `www.google.com` from the `google.com` servers; earlier it learned the address of the `google.com` servers from the `.com` servers; earlier it learned the address of the `.com` servers from the Internet's central "root DNS servers". Each address is cached for some time, reducing the overall load on the root DNS servers to 120 billion queries per day (according to [106]), around 1.4 million queries per second. These servers have 13 names (root server A, root server B, and so on through root server M), but according to [103] there are actually "hundreds" of physical computers; if "hundreds" means 200 then an average root server handles around 7000 queries per second.

A 2021 statement [104] from the root operators indicates that these servers will not deploy encryption for now. The primary reason stated is performance:

> Due to the critical role that root name servers play, combined with the fact that they are themselves often targets of DDoS attacks, Root Server Operators have some concerns about supporting DNS encryption for serving the root zone. It is well known that UDP has desirable performance characteristics, due to its stateless nature. Increasing the state-holding burden with the addition of connection-oriented protocols, as well as encryption data, not only reduces the performance of name servers, but also may raise new types of denial-of-service attacks.

It is certainly true that running DNS over a stateful HTTPS connection, rather than over stateless UDP, opens up denial-of-service attacks; see, e.g., [29, Section 2.2]. But it is simply not true that DNS encryption needs connections. DNSCurve, a simple DNS transport layer that uses X25519 for encryption and authentication, was already deployed by OpenDNS in 2010, according to [91]; there are faster DH proposals, but X25519 suffices for the following analysis. DNSCurve runs over UDP in the same way that DNS normally does, rather than requiring the server to accept connections and store state for each connection.

Regarding the impact of encryption on CPU load, the software from [85] takes 95437 Skylake cycles for X25519, and all other operations in DNSCurve are much faster. The software performs more than 125000 X25519 operations per second on a low-cost quad-core 3GHz Intel Xeon E3-1220 v5 CPU from 2015. If a root server is using such a CPU to handle its 7000 DNS queries per second, and those queries all upgrade tomorrow to DNSCurve, then X25519 will consume under 6% of the available CPU time. Presumably most root CPUs are more powerful than this; on a newer, higher-cost server with two AMD EPYC 7742 CPUs from 2019, the same software performs more than 2.9 million operations per second.

Regarding network traffic, a DNS query for `.com` today is 21 bytes plus 8 bytes of UDP overhead, 20 bytes of IP overhead, and 38 bytes of Ethernet overhead;

the response is 509 bytes (without IPv6) plus the same overheads. DNSCurve makes each query 68 bytes longer—there is an 8-byte protocol selector, a 32-byte X25519 key, a 12-byte nonce, and a 16-byte MAC—and each response 48 bytes longer. This 17.5% difference in traffic volume is barely noticeable compared to the much larger network capacity already available in the root servers. To quantify this, consider the 2015 denial-of-service attack described in [102], an attack that "saturated network connections" for *some* of the root servers but still did not take down root DNS service. This attack used "up to approximately 5 million queries per second, per DNS root name server letter receiving the traffic", and hit "most" of the 13 letters—tens of millions of queries per second overall. Presumably the root servers have even more bandwidth today.

What happens if, instead of trying to deny service by flooding a network, an attacker tries to deny service by flooding a CPU with X25519 operations? The software mentioned above handles 750 megabits per second of X25519 keys on the EPYC server. This corresponds to more than 3 gigabits per second of DNS network traffic, given minimum packet overheads. Sites with larger Internet connections can spread the load across multiple servers to guarantee that the network running at full capacity cannot overload the CPUs with cryptographic operations. Sites with smaller Internet connections can use smaller servers.

2.2. The machinery of cryptographic performance advertising, part 2: confirmation bias. Usually the factual basis for cryptographic performance advertising is much less clear. Here's a case study.

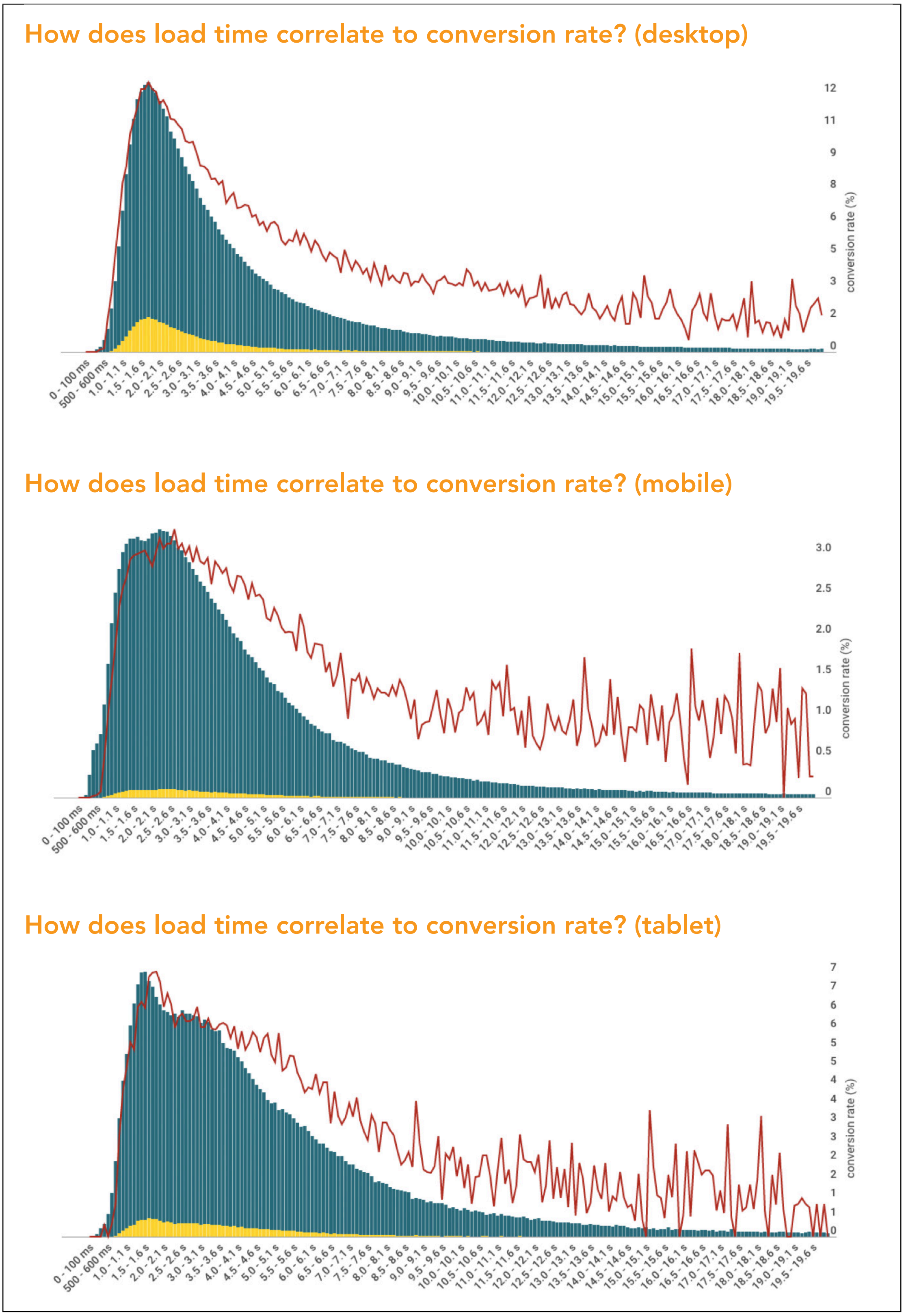
The 2020 paper "Post-quantum authentication in TLS 1.3: a performance study" [110] states "Reports like [1] lead us to believe that hundreds of extra milliseconds per handshake are not acceptable". The cited document "[1]" is a 2017 Akamai press release "Akamai online retail performance report: milliseconds are critical" [6], subtitled "Web performance analytics show even 100-millisecond delays can impact customer engagement and online revenue".

Akamai's underlying report [7] says, as one of its "key insights", that "just a 100-millisecond delay in load time hurt conversion rates by up to 7%". My understanding is that "conversion" has the following meanings:

- in traditional sales terminology, converting a potential customer into a lead (i.e., contact information for the potential customer) or a sale;
- in web sales terminology, converting a view of a product web page into a sale of the product.

A reader who digs into the report finds a statement that "desktop pages" that loaded "100 milliseconds slower" experienced a "2.4% decrease in conversion rate"; and a statement that for smartphones 2.4% changes to 7.1%. Apparently these statements are the source of the "key insight" stating "up to 7%".

Let's look at the underlying data more closely. Akamai hosts its customers' web pages, caching copies of those pages on thousands of Akamai-run computers around the world, the idea being that browsers will receive each page quickly from a nearby computer. The report says that Akamai collected metadata on billions of web sessions from "leading retail sites" that are Akamai customers



How does load time correlate to conversion rate? (desktop)

How does load time correlate to conversion rate? (mobile)

How does load time correlate to conversion rate? (tablet)

Fig. 2.3. Screenshot from [7, page 8].

(with permission from the sites). In Figure 2.3 here, a screenshot from [7, page 8], the yellow area shows the distribution of page-load times in converted web sessions, the blue area shows the same for non-converted web sessions, and the red curves show the percentage of web sessions that were converted.

For example, for desktop browsers (top graph), about 12% of sessions were converted for page-load times around 1.8 seconds: at horizontal position 1.8, the red curve is around 12%, and the top of the yellow is about 12% of the top of the blue. The conversion rate drops off for larger page-load times: e.g., about 6% of sessions were converted for page-load times around 4 seconds, meaning that the conversion rate was 50% smaller. The report highlights the conversion rate being 12.8% at 1.8 seconds, and 2.4% smaller (i.e., 12.5%) at 1.9 seconds, the conclusion being that a 100-millisecond delay in load time is measurably bad.

Notice, however, that the conversion rate in the graph also drops off for *smaller* page-load times. Compare the following numbers:

- ≈6% of sessions were converted for page-load times around 1.1 seconds.
- ≈8% of sessions were converted for page-load times around 1.2 seconds.
- ≈9% of sessions were converted for page-load times around 1.3 seconds.
- ≈12% of sessions were converted for page-load times around 1.8 seconds.

Why did Akamai's report not conclude that a 100-millisecond delay in load time is measurably *good*?

The report briefly addresses this question, saying "the faster end of the bell curve is primarily comprised of 404 pages and other pages that, while fast, do not fall on the conversion path". But wait a minute. If different types of web pages can fail to produce sales, explaining the low end of the graph, then can't such differences explain the *entire* graph? Maybe the situation is that there are

- "too simple" product pages (which tend to load quickly) that don't have enough pictures to convince the typical customer, and
- "too complex" product pages (which tend to load slowly) that scare the typical customer away, and
- "just right" product pages (which tend to have load times in the middle) that do the best job of attracting customers.

Could it be that what matters for sales isn't actually the last bit of speed in delivering the web page, but rather the content of the web page?

Another section of the same report says that "user sessions that converted contained 48% more scripts than sessions that did not convert". Perhaps a three-dimensional analysis of conversion, scripts, and load times would show that load times don't matter when the number of scripts is the same. Perhaps there are other confounding factors, such as lower-income customers buying fewer products and also tending to have slower network connections.

Recall that Akamai's report portrays a 100-millisecond delay as being worse for smartphones than for desktops, losing 7.1% of the smartphone conversions. But the smartphone conversion rate in Figure 2.3 (middle red graph) drops off *more gently* than the desktop conversion rate. For example, the smartphone conversion rate is 3.3% at 2.7 seconds, and is half of 3.3% around 6 seconds. Akamai's report seems to be alluding to the first drop in the red graph after its peak, but the red graph then jumps up a moment later, and in general the small-scale wobbling in the red graph looks like random noise.

Maybe a properly designed study that adds 100 milliseconds of delay into web pages for randomly selected users would produce a 2%, maybe even 7%, drop in sales. But the study in [7] was not properly designed. There are many other theories that would produce the graphs in [7]. The highlighting of one *possible* theory is easily explained as a combination of confirmation bias and marketing.[3] The same report briefly mentions that "Walmart saw up to a 2% increase in conversions for every second of improvement in load time", without noting that, compared to 7% for 100 milliseconds, 2% for 1 second is 35× less important.

2.4. The machinery of cryptographic performance advertising, part 3: systematic exaggeration. Let's return to the belief in [110] that "hundreds of extra milliseconds per handshake are not acceptable". Most of the page loads in Figure 2.3 are at least hundreds of milliseconds beyond the red peaks, and yet these are, according to Akamai, deployed web pages from "leading retail sites"; so how does [110] conclude that this extra time is "not acceptable"?

Even if hundreds of extra milliseconds per page load are unacceptable, it is an error to conflate this with hundreds of extra milliseconds *per handshake*. A TLS handshake sets up a session that can be, and often is, used to load many pages without further handshakes. A page often collects content from multiple servers, but presumably most latencies overlap. Perhaps users wouldn't actually notice the costs considered in [110]—or perhaps they would notice the costs but would consider the costs *acceptable* if sufficient benefit is provided.

Google constantly makes changes to its Chrome browser. Sometimes these changes improve performance. Sometimes they reduce performance. Google has a statement [60] of procedures for deciding whether a performance regression is acceptable. This statement spends several paragraphs listing "some common justification scenarios", such as the following:

> **What do we gain?** It could be something like: . . . Additional security

Compare this to a November 2021 Cloudflare blog post [124] pointing to the same statement of Google procedures and claiming that "only in exceptional cases does Chrome allow a change that slows down any microbenchmark by even a percent". This claim plays a pivotal role in the advertising for [124], which, like [110], studies the performance of signature systems.

All of the signature systems listed in [110, Table 1] have software available that signs in under 20 milliseconds on a 3GHz Intel Haswell CPU core from 2013 (never mind the possibility of parallelizing the computation across several cores). The server signs a TLS handshake only once. The browser also verifies certificate

[3] I'm not trying to say that Akamai's many customers are making a mistake. On the contrary: my impression is that Akamai is providing robust web service to its customers, and at the same time is doing a valuable public service in allocating Internet links more efficiently.

chains—the paper considers TLS sessions with 3 or 4 verification operations—but all of these signature systems have software available that verifies in under 5 milliseconds. The total size of a public key and a signature in [110, Table 1] is at most 58208 bytes, so a 100Mbps network connection (already common today, never mind future trends) can transmit 4 public keys and 4 signatures in 20 milliseconds. For comparison, [67, "Total Kilobytes"] shows that the median web page has grown to 2MB, and that the average is even larger.

Given the numbers, it is hard to see how TLS users will care which of these signature systems is used, and it is hard to see why one should care about a more detailed performance study. The paper [110] thus has an incentive to paint a different picture. The paper starts from the "not acceptable" belief quoted above; selects signature-system software that was "not optimized"; configures a server-controlled networking parameter, `initcwnd`, to wait for a client reply after sending just 15KB of data; and devotes a page to discussing long-distance connections, such as a US-to-Singapore connection, where waiting for a client reply costs a 225-millisecond round trip.[4]

The server's `initcwnd` sheds light on the question of how important latency is. In the original TCP congestion-control algorithms from the late 1980s [70], a server with any amount of data to send begins by sending at most one packet to the client. The server then waits for a client reply, then sends a burst of two packets, then waits for a client reply, then sends a burst of four packets, etc. The initial packet is allowed to be full size, which today typically means 1500 bytes. The choice to start with one full-size packet, not more and not less, is a balance between (1) the slowdown from delaying subsequent data and (2) concerns that having everyone start with more data would overload the network.

In 1998, an "Experimental" RFC [10] proposed increasing the server's "initial congestion window" (`initcwnd`) from 1 packet to 2–4 packets. Today this is a "Proposed Standard" [11]. In 2013, another "Experimental" RFC [41] proposed increasing `initcwnd` to 10 packets. According to [39], Akamai was using 16 packets in 2014, and 32 packets in 2017. Five doublings of `initcwnd`, from 1 to 2 to 4 to 8 to 16 to 32, eliminate four or five round-trip times from any sufficiently large file transfer.

Other major web servers in 2017 used `initcwnd` ranging from 10 through 46, according to [39]. All of these are still officially "experimental", far above the standard 1 and the proposed standard 2–4, but most Internet links have grown to handle massive video traffic, and a relatively tiny burst of packets at the beginning of a TCP connection does not cause problems. Meanwhile [110] refuses to benchmark `initcwnd` above 10, and issues an ominous warning that widespread deployment of a larger `initcwnd` "could have adverse effects on TCP Congestion Control", as if a larger `initcwnd` were not already widely deployed.

In the opposite direction, compared to a web server taking `initcwnd` as 46, a web server taking `initcwnd` as just 10 is sacrificing two round trips, almost half a second for a connection between the US and Singapore. If such a slowdown

[4] The blog post [124] instead measures Cloudflare's usual short-distance connections. Unsurprisingly, the observed times are much smaller than in [110].

is “not acceptable” then why are some major web servers doing it? Perhaps the answer is that such long-distance connections are rare. Or perhaps the answer is that occasional delays of hundreds of milliseconds aren’t actually so important.

2.5. Competitions for cryptographic performance. There is overwhelming evidence of performance requirements—whether real or imagined—playing an important, perhaps dominant, role in cryptographic competitions:

- At an NBS workshop in 1976, before DES was approved as a standard, Diffie (in joint work with Hellman; see [50, page 77, “Cost of larger key”]) proposed modifying the DES key schedule to use a longer key. Representatives of Collins Radio and Motorola objected to this proposal, saying that DES is “close to the maximum that could be implemented on a chip with present technology” and that a manufacturing delay “of one to two years might be encountered if a longer key were required”. See [84, page 20].[5]
- The AES call for submissions [77] listed “computational efficiency” as the second evaluation factor after “security”. NIST’s final AES report [89, page 528] stated that “Rijndael appears to offer an adequate security margin” and that “Serpent appears to offer a high security margin”, and the same report claimed [89, page 516] that “security was the most important factor in the evaluation”, but NIST selected Rijndael rather than Serpent as AES. NIST’s only complaints about Serpent were performance complaints.
- eSTREAM called [55] for “stream ciphers for software applications with high throughput requirements” and called for “stream ciphers for hardware applications with restricted resources such as limited storage, gate count, or power consumption”. The eSTREAM committee selected several ciphers for the final eSTREAM portfolio [14]—for example, selecting my Salsa20 cipher with 12 rounds, “combining a very nice performance profile with what appears to be a comfortable margin for security”. I had recommended, and continue to recommend, 20 rounds; I had proposed reduced-round options only “to save time” for users “who value speed more highly than confidence”.
- The SHA-3 call for submissions [74] said that “NIST expects SHA-3 to have a security strength that is at least as good as the hash algorithms currently specified in FIPS 180–2, and that this security strength will be achieved with significantly improved efficiency”. When NIST selected Keccak as SHA-3, it wrote [40] that Keccak “offers exceptional performance in areas where SHA-2 does not”, and that Keccak “received a significant amount of cryptanalysis, although not quite the depth of analysis applied to BLAKE, Grøstl, or Skein”. On the other hand, NIST’s prioritization of efficiency over security was not as

[5] NBS wrote in 1977 [49, page 10] that DES was “satisfactory for the next ten to fifteen years as a cryptographic standard”. NIST did not end up withdrawing DES as a standard until 2005. Almost all of the DES benefits claimed in [79] appeared in 1980 or later, as did 93% of the implementations listed in [79, page 31]. Why was a claimed manufacturing delay from 1976 to 1977 or 1978 treated as important? See Section 3.6 for a possible answer. It is also far from clear that the claim was correct: [79, page 16] reports that IBM “had developed a commercially viable VLSI chip that could incorporate the encryption algorithm efficiently” already before March 1975.

clear for SHA-3 as for AES: [40] also said that Keccak "relies on completely different architectural principles from those of SHA-2 for its security".
- CAESAR called [18] for "authenticated ciphers that (1) offer advantages over AES-GCM and (2) are suitable for widespread adoption". Proposals varied in whether they emphasized security advantages or efficiency advantages. The CAESAR committee later identified three "use cases" [20]: "lightweight applications" requiring "small hardware area and/or small code for 8-bit CPUs"; "high-performance applications" requiring "efficiency on 64-bit CPUs (servers) and/or dedicated hardware"; and, deviating from the speed theme, "defense in depth" providing "authenticity despite nonce misuse". Ultimately the CAESAR committee selected Ascon (first choice) and ACORN (second choice) for use case 1, AEGIS-128 and OCB (without an order) for use case 2, and Deoxys-II (first choice) and COLM (second choice) for use case 3.
- The most recent NISTPQC report [9] refers repeatedly to performance as a basis for decisions. For example, the report says that NIST's "first priority for standardization is a KEM that would have acceptable performance in widely used applications overall. As such, possible standardization for FrodoKEM can likely wait". What is not "acceptable" in Frodo's performance? NIST writes that a TLS server using Frodo uses "close to 2 million cycles" and "receives a public key and a ciphertext (around 20,000 bytes in total) for every fresh key exchange".[6]
- NISTLWC called [87] for submissions of hash functions and authenticated ciphers "tailored for resource-constrained devices", since "many conventional cryptographic standards" are "difficult or impossible to implement" in such devices, at least with the constraint of acceptable performance. This is the most recent competition in this list, with an initial submission deadline in 2019.

Almost all of these competitions are for symmetric cryptography: block ciphers, hash functions, authenticated ciphers, etc. Symmetric cryptography is applied to *every byte of communicated data*, authenticating every byte and encrypting every potentially confidential byte. The Android storage-encryption example (almost) fits this pattern, with every byte encrypted (but currently not authenticated) before being communicated to the untrusted storage device. As another example, the fastest signature systems involve

- hashing all the data being signed;
- some asymmetric work independent of the data length.

Similarly, a TLS session applies an authenticated cipher to every byte of data, after an asymmetric handshake independent of the data length. Trends towards larger data volumes, supported by faster CPUs and faster networks, mean that a larger and larger fraction of overall cryptographic cost comes from symmetric

[6] One side would send a 9616-byte public key, and the other side would send back a 9720-byte ciphertext, so "receive" is not true. It's true that each side would use close to 2 million Haswell cycles: i.e., close to a full *millisecond* on a 2GHz CPU core.

cryptography, providing one reason for having more symmetric competitions than asymmetric competitions.

Perhaps more advanced cryptographic operations will dominate cryptographic costs someday. The yearly iDASH "secure genome analysis competition" [114] measures performance of homomorphic encryption, multiparty computation, etc. As another example, to the extent that post-quantum cryptography is more expensive than pre-quantum cryptography, it changes the balance of costs—which, again, does not imply that its costs matter to the end users; this is something that needs analysis.

Comparing the symmetric competitions shows trends towards larger and more complex inputs and outputs in the cryptographic algorithm interfaces. DES has a 64-bit block size; AES has a 128-bit block size. Stream ciphers encrypt longer messages. Hash functions hash longer messages. Authenticated ciphers include authentication tags in ciphertexts, and optionally authenticate another input. Many NISTLWC submissions support hashing and authenticated encryption, sharing resources between these functions. This does not mean that complexity is a goal per se: symmetric algorithms with larger interfaces often reach levels of efficiency that seem hard to achieve with smaller interfaces, and there are some security arguments for larger interfaces. See generally [21, Section 2].

2.6. How AES performance was compared. During the AES competition, Biham [33, Table 3] reported that "the speed of the candidate ciphers on Pentium 133MHz MMX" was 1254 cycles for Twofish, 1276 cycles for Rijndael, 1282 cycles for CRYPTON, 1436 cycles for RC6, 1600 cycles for MARS, 1800 cycles for Serpent, etc. for encrypting a 128-bit block under a 128-bit key.

Serpent, generally viewed as the AES runner-up, had (and has) a much larger security margin than Rijndael, the eventual AES winner. Biham also reported speeds scaled to "proposed minimal rounds": 956 cycles for Serpent (17 rounds rather than 32), 1000 cycles for MARS (20 rounds rather than 32), 1021 cycles for Rijndael (8 rounds rather than 10), etc.

Let's focus on Biham's reported 1276 cycles for full 10-round Rijndael, almost 80 cycles per byte. The Pentium (with or without MMX) could run at most 2 instructions per cycle, and obviously it didn't have AES instructions, but did it really need 1276 cycles for 160 table lookups and some auxiliary work? No, it didn't. Schneier, Kelsey, Whiting, Wagner, Hall, and Ferguson [107, Table 2] reported Rijndael taking 320 Pentium cycles, just 20 cycles per byte. They also estimated that Serpent would take 1100 Pentium cycles—but then Osvik [92, page 8] reported an implementation taking just 800 cycles.

Compared to Biham's reports, Serpent was more than 2× faster, and Rijndael was 4× faster. Overall these speedups seem to favor Rijndael, for example putting Rijndael ahead of Serpent in the "proposed minimal rounds" speed. On the other hand, overall these speedups *compress* the difference in costs between Serpent and Rijndael, from $1800 - 1276 = 524$ cycles to $800 - 320 = 480$ cycles; and these speedups make it more likely that both ciphers will meet the users' performance requirements.

Why did these reports end up with such different numbers? And why did NIST's AES efficiency testing [86] feature CRYPTON as the fastest candidate in its tables and its graphs, 669 Pentium Pro cycles to encrypt, with Rijndael needing 809 Pentium Pro cycles to encrypt? The Pentium Pro is generally faster than the Pentium (and Pentium MMX); [107] reported 345 Pentium Pro cycles for CRYPTON and 291 Pentium Pro cycles for Rijndael.

2.7. The process of comparing cryptographic speeds. All of these speed numbers arise from the general process shown in Figure 2.8. The first column has a cryptographic algorithm: for example, the Rijndael encryption algorithm mapping a 128-bit plaintext and a 128-bit key to a 128-bit ciphertext. The second column has a programmer writing software for this algorithm—or for another algorithm computing the same mathematical function. The third column has a benchmarking mechanism that measures the speed of the software, for example the number of cycles that the software uses on a 133MHz Pentium MMX CPU. The fourth column has an advertisement mechanism that might or might not bring the resulting cycle count to the attention of readers.

There are several reasons that the outputs of this process vary:

- The cryptographic functions vary: e.g., Rijndael and Serpent have different speeds. The whole point of a speed competition is to compare the speeds of different functions.
- The CPUs vary. This complicates comparisons. If function F is faster than function G on one CPU, but slower on another, then which function wins the competition?
- The software varies. A programmer often fails to achieve the best speed for a cryptographic function on a CPU. The slowdown depends on many details of the function, the CPU, the programmer's experience, and the programmer's level of enthusiasm for the function. There are many counterexamples to the notion that the slowdown is independent of the function: for example, compared to [107], NIST's study [86] slowed down CRYPTON by a factor 1.94, slowed down Rijndael by a factor 2.78, and reversed the comparison between the algorithms.
- The benchmarking mechanism varies, for example in the handling of per-input timing variations, initial code-cache-miss slowdowns, operating-system interrupts, clock-frequency variations, and cycle-counting overheads.
- The advertisement mechanism varies. As an example, measurements that are slower than previous work are likely to be suppressed if the advertisement mechanism is a paper claiming to set new speed records, but less likely to be suppressed if the advertisement mechanism is a paper claiming to compare multiple options.

Both 1276 cycles from [33] and 320 cycles from [107] are reported to be Rijndael measurements on the Pentium, so the first and second effects cannot explain the gap. The easiest explanation is the third effect, although it is easy to imagine some contributions from the fourth and fifth effects.

The situation is different when cryptographic functions are deployed. The CPUs still vary, but, for each CPU, slower software is systematically suppressed

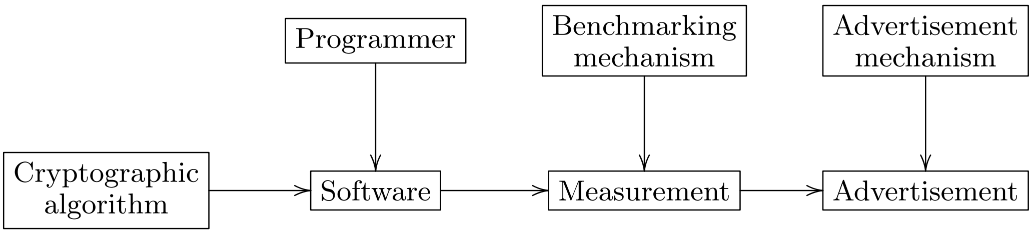


Fig. 2.8. The process of producing performance data for cryptographic software.

in favor of faster software (as measured by a unified benchmarking mechanism), because users who care about speed don't want the slower software. For example, the OpenSSL cryptographic library contains 26 AES implementations, almost all in assembly language, in a few cases weaving AES computations together with common hash-function computations. The library checks the target platform and selects an implementation accordingly.

When different implementations run at different speeds for the same function on the *same* CPU, what speed does a speed competition assign to that function? Here are two strategies for answering this question:

- The **"fair and balanced" strategy** gives equal weight to the speeds of all implementations.
- The **real-world strategy** takes the fastest available implementation, the same way that a cryptographic library does, while suppressing the speeds of slower implementations.

As soon as there are two implementations running at different speeds, the "fair and balanced" strategy reports worse speed than the real-world strategy, speed farther from what the users will see (assuming the users care about speed). The real-world strategy creates a healthy incentive for implementors to look for and eliminate slowdowns in their own implementations, while the "fair and balanced" strategy creates an unhealthy incentive for implementors to "accidentally" create slow implementations of competing functions.

The standard argument for the "fair and balanced" strategy is to say that reducing CPU time is rarely worth the software-development time. Knuth [76, page 268] famously expressed the tradeoff as follows:

> Programmers waste enormous amounts of time thinking about, or worrying about, the speed of noncritical parts of their programs, and these attempts at efficiency actually have a strong negative impact when debugging and maintenance are considered. We *should* forget about small efficiencies, say about 97% of the time; premature optimization is the root of all evil.

But Knuth's complaint here is about optimizing "noncritical parts" of programs. Knuth continued as follows:

> Yet we should not pass up our opportunities in that critical 3%. A good programmer will not be lulled into complacency by such reasoning, he

> will be wise to look carefully at the critical code; but only after that code has been identified.

Today there are tens of millions of lines of code in a web browser, and many more lines of code in a complete computer system. Almost all of this code disappears if one asks which code has a noticeable impact on performance. A few “hot spots” are so important that implementors look carefully at making them run as quickly as possible. If a cryptographic operation is not one of these “hot spots”, then why is it the topic of a speed competition?

2.9. How AES speeds were compared, part 2. Say someone publishes a faster implementation of a cipher, 500 lines of software, two years after a cipher competition begins. Does this mean that the software took 24 person-months of work, and that similarly optimizing for the 20 most important platforms would take 480 person-months of work? Or could it be that the same person was busy with multiple projects, spending only two months of work on this project, and could it be that a new platform shares 75% of the optimization work with previous platforms, so optimizing for the 20 most important platforms would take under 12 person-months of work? Even if it’s really 480 person-months, wouldn’t the community invest this effort in any widely deployed cipher? Compare [80], which estimates that AES produced $250 billion in worldwide economic benefits between 1996 and 2017.

NIST’s AES report [89] did not measure AES code-development time but claimed that this time was often important:

> In some environments, the speed at which the code runs is perceived as a paramount consideration in evaluating efficiency, overriding cost considerations. In other cases, the time and/or cost of code development is a more important consideration.

NIST deviated slightly from the “fair and balanced” strategy, and in particular refused to list speeds of the fast Serpent implementations from [92] and [58], since those implementations had been constructed by “1000 hours of execution of search programs” and “do not necessarily port to different platforms”.

Around that time the Internet was reported to be communicating roughly 2^{58} bytes per year, more than doubling every year. The load was spread across millions of CPUs. It is hard to see how a few dozen CPUs spending a day on “execution of search programs” can be a cost to worry about, even if one repeats those hours for dozens of different target platforms. Furthermore, it is easy to see that the optimizations from [92] and [58] work reasonably well on a wide range of platforms, even if further searching would do better on some platforms.

It is important to realize the mismatch between the resources available for widely deployed algorithms and the resources available *during* a competition. The costs of cryptographic optimization—in human time or computer time—can be huge obstacles for submitters without serious optimization experience. The submitters ask for help from people with experience, but the only people paying attention at this point are crypto junkies, and there are many submissions. Any particular algorithm struggles to gain attention. It is easy to see how this struggle could be misinterpreted as a reflection of the deployment environment, rather than as a problem created by the competition process.

Think about the cryptographic function that would win a speed competition if it were properly optimized. There is a risk that this function loses the competition because the necessary optimizations are not demonstrated in time. Ways to reduce this risk include

- specifying a small number of target platforms as the arenas for competition, to better focus optimization work during the competition (although there is then a risk that these platforms are inadequate representatives of other platforms);
- tracking expert assessments of the unexplored avenues for speedups in each submission; and
- extending the competition time until the performance picture has settled down.

The "fair and balanced" strategy exacerbates this risk by assigning a low weight to speedups found later in the competition, whereas the real-world strategy ignores all worse speeds the moment that better speeds have been demonstrated.

One might think that Rijndael was much faster than Serpent even after the speedups, so assigning higher weights to the speedups could not have changed the AES selection. But the speed picture was not this simple. The hardware data surveyed in [89] suggested that the most efficient proposal for hardware encryption was pipelined counter-mode encryption using Serpent.[7] Equalizing security margins would have made Serpent much faster. Rijndael was faster in software because its 256-entry table lookups reused existing CPU instructions—but hardware implementations have to pay for table lookups. The optimizations from [92] and [58] should have increased Serpent's hardware advantage while decreasing its software disadvantage. If I've counted correctly then [58] uses 201 bit operations per plaintext bit for full 32-round Serpent (plus 1 bit operation per plaintext bit for counter-mode xor and a small cost for counter maintenance), very much like the best operation count known today for 10-round Rijndael.

2.10. Better benchmarking mechanisms. NESSIE, mentioned in Section 1, was a 2000–2003 EU project "New European Schemes for Signatures, Integrity, and Encryption". I'm not sure NESSIE qualifies as a competition—it selected 17 algorithms—but in any case it took important steps towards matching its performance evaluations with the reality of what cryptographic users would see. NESSIE published a software API supporting secret-key encryption, public-key signatures, etc.; collected C implementations of many cryptographic functions, with all implementations using the same API; tuned the C implementations for speed; wrote a benchmarking toolkit to measure speed; ran the benchmarking toolkit on many computers; and published the results. See [97].

[7] There were complaints about Serpent using extra hardware for the inverse function, but counter-mode encryption does not need the inverse.

As part of the eSTREAM competition, Christophe De Cannière developed a new API for stream-cipher software, and wrote a new benchmarking toolkit [**36**] to measure implementations supporting the API. This toolkit was limited to stream ciphers but had several advantages over the NESSIE toolkit. Notably, it tried more compiler options; it supported assembly-language software; and it was *published*. Implementors could run the toolkit to quickly and reliably see how fast their own software was, and to guide improvements to the software. Third parties could run the toolkit to contribute and verify public benchmark results. The quick feedback to implementors—from running the toolkit on their own machines and from seeing results announced by third parties—led to many implementation improvements during eSTREAM. The toolkit followed the real-world strategy, automatically reporting a list of ciphers with the speed of the fastest implementation of each cipher; see [**36**, Section 6].

In its final AES report [**89**, Section 2.5], NIST had complained that requests to consider a different number of rounds for an AES submission "would impact the large amount of performance analysis" that had already been done, since "performance data for the modified algorithm would need to be either estimated or performed again". It wasn't realistic to expect all the authors of performance-comparison papers to integrate new functions into their private benchmarking procedures and update their papers accordingly. This complaint goes away when the benchmarks come from an easily extensible *public* benchmarking toolkit: anyone can tweak the number of rounds in the implementations and run the toolkit again.

In 2006, Tanja Lange and I started eBATS, a new benchmarking project for public-key systems. Together with Christof Paar, Lange was the leader of the Virtual Application and Implementation Research Lab, VAMPIRE, within a European network, ECRYPT; the name "eBATS" stands for "ECRYPT Benchmarking of Asymmetric Systems". STVL, ECRYPT's Symmetric Techniques Virtual Lab, was running eSTREAM.

Lange and I designed a simple new cryptographic API to handle the needs of benchmarking *and* the needs of cryptographic libraries, so writing software for benchmarking was no longer inherently a separate task from writing software for real-world use. This increased the incentive for implementors to support the benchmarking API, and decreased the extra implementation effort. We analyzed and improved the end-to-end benchmarking process that turned new software into the public presentation of measurements. Extra feedback to implementors added extra incentives to contribute implementations.

In 2008, eSTREAM was drawing to a close, and the SHA-3 competition had been announced. Lange and I started eBACS, a unified benchmarking project that includes eBASC for continued benchmarking of stream ciphers, eBASH for benchmarking of hash functions, and eBATS. We replaced BATMAN, the original benchmarking toolkit for eBATS, with a new benchmarking toolkit, SUPERCOP. By late 2009, eBASH had collected 180 implementations of 66 hash functions in 30 families.[8] eBASH became the primary source of software-performance information for the SHA-3 competition. See [40].

eBACS has continued since then, adding more cryptographic functions, more implementations of those functions, and newer CPUs—while continuing to run benchmarks on years of older CPUs for comparability. The SUPERCOP API was carefully extended to handle more operations, such as authenticated encryption. CAESAR, NISTPQC, and NISTLWC required submissions to provide software using the SUPERCOP API. SUPERCOP now includes 3716 implementations of 1255 cryptographic functions in hundreds of families. See [28].

This is not the end of the story. SUPERCOP measures cryptographic speeds on CPUs large enough to run Linux, but what about microcontrollers? FPGAs? ASICs? Performance metrics other than speed, such as energy usage? Notable efforts to improve benchmarking processes include the ongoing ATHENa project for FPGAs and ASICs, and the XBX, FELICS, XXBX, FELICS-AEAD, and `pqm4` projects for microcontrollers. See generally [57], [121], [122], [51], [38], [53], [73], and [72].

3 Security

Here we are, halfway through a paper that claims to be analyzing the extent to which competition procedures reduce security risks, and all I've been talking about is speed competitions. This section closes the gap.

3.1. The complex relationship between speed and security. Let's begin with the obvious argument that a cryptographic speed competition *is* a security competition.

Risk #1 of cryptography is that the cryptography isn't used. One reason that cryptography isn't used, as mentioned in Section 1 and illustrated by the 4-year delay in Android encryption reviewed in Section 2.1, is that the cryptography doesn't meet the user's speed requirements. Perhaps the requirements are driven by reality—something with worse efficiency would, if deployed, be a problem—or perhaps they are driven by *fear* that there will be a problem. Either way, something that doesn't meet the requirements won't be deployed, and if nothing meets these requirements then nothing will be deployed. A cryptographic speed competition identifies the functions that have the best chance of meeting the user's speed requirements.

There is, however, an equally obvious argument in the opposite direction, namely that a cryptographic speed competition naturally identifies the *weakest* functions. RSA-1024 is more efficient than RSA-2048, and RSA-512 is more efficient than RSA-1024, so RSA-512 will beat RSA-1024 and RSA-2048 in a speed competition—but RSA-512 is breakable. AES candidates were slower with 256-bit keys than with 128-bit keys. Rijndael proposed not just multiple key sizes

[8] I don't mean to suggest that "family" has a clear definition here. Is SHA-1 in the same family as SHA-224, SHA-256, SHA-384, and SHA-512? The benchmarking process measures each function separately.

but also multiple block sizes, with top speed requiring the minimum block size. DES had just one version, but Diffie and Hellman proposed a longer-key variant, and people complained that this was more expensive; see Section 2.5.

If each family of algorithms claims a tradeoff between security and efficiency, then it is unsurprising that graphing the claimed security and efficiency of many different proposals will also show such a tradeoff; see, e.g., the general slant of the graphs in [22]. How, then, is a competition for top speed not the same as a competition for minimum security? The users will have something that meets their performance requirements—something breakable.

Most competitions respond by specifying a **minimum allowed security level**. A more subtle extension of this response is to say that users should take the **maximum security margin**. This works as follows:

- Identify the most efficient cryptographic function that seems to meet or exceed the minimum allowed security level. This is a speed competition, but subject to a security requirement.
- Within the family containing the most efficient function, take the *largest* function that meets the users' performance requirements.

The idea here is that the speed competition gives users the maximum room for larger keys, more cipher rounds, and other forms of security margins that—we hope—provide a buffer against attack improvements.

For example, say the efficiency metric is bit operations per bit of plaintext to encrypt a long stream; and say the minimum allowed security level is 2^{128}. My understanding of current attacks is that Serpent reaches this security level with 12 rounds, using about 75 operations per bit; Rijndael reaches this security level with 8 rounds, using about 160 operations per bit; and Salsa20 reaches this security level with 8 rounds, using 54 operations per bit. If these are the competitors then Salsa20 wins the speed competition.[9] A user who can afford, say, 80 operations per bit then takes 12 rounds of Salsa20 (78 operations per bit). The same user would also be able to afford 12 rounds of Serpent, but 12 rounds of Salsa20 provide a larger security margin, presumably translating into a lower risk of attack.

The reality, however, is that cryptographic designers are overconfident, and see negligible value in large security margins ("I can't have missed something so big"), even when history shows one example after another of attacks that would have been stopped by large security margins. Meanwhile the same designers see that proposing something with a large security margin is risky. Serpent proposed more than twice as many rounds as necessary and had the whole proposal dismissed as being too slow.

I knew that Salsa20 had far more rounds than I could break, correctly guessed that it had far more rounds than anyone else could break, correctly guessed that

[9] This is an unfair comparison. Salsa20 was designed years later, taking advantage of lessons learned from Serpent and many other designs. Salsa20 also benefits from spreading differences through a 512-bit block, while the AES competition required 128-bit blocks and discarded Rijndael's 256-bit-block options.

it would be competitive in speed anyway, and concluded that it would be able to get away with a large security margin. At the same time, knowing what had happened with Serpent, I didn't want to go beyond 20 rounds. I held off on proposing reduced-round versions: an initial proposal with (say) 12-round and 20-round options would have been interpreted as indicating a lack of confidence in 12 rounds, whereas an initial proposal of 20 rounds followed by "Look, people can't break 12 rounds, and can't even break 8 rounds" sounded purely positive. There was an eSTREAM requirement to support 128-bit keys, but I proposed as many rounds for 128-bit keys as for 256-bit keys, so that users wouldn't have a speed incentive to take smaller keys.

All of this was converting a presumed software-speed advantage into extra security margin—but if someone else had designed a similar stream cipher with a smaller round-count parameter then Salsa20 would have been eliminated. In the opposite direction, there is a long history of submissions being eliminated from competitions because they chose parameters *slightly* too small for security—even if larger parameters would have been competitive. Is a competition supposed to be evaluating the best tradeoffs available between speed and security, or is it supposed to be evaluating the designer's luck in initial parameter selection?

Submitters see what happened in previous competitions. This feedback loop keeps most security margins in a narrow range. Designers and implementors don't want to risk making mistakes in providing larger options. In the end, users aren't being given the choice to take the largest security margin they can afford. I did convince the relevant people that TLS would be just fine in performance using 20 rounds of ChaCha rather than 12, but most deployed security margins are much smaller than this, and competitions follow suit.

3.2. The complex relationship between speed and security, part 2: later discovery of attacks. If we've correctly evaluated the security level of a cryptographic algorithm, and if that security level is high enough compared to the resources available to the attacker, then we shouldn't need a security margin—the algorithm is secure. The basic problem here is that we don't have procedures to reliably evaluate the security level of a cryptographic algorithm. Sometimes breaking an algorithm takes years or even decades of public attack development. This does not mean that the algorithm was secure in the meantime: the algorithm was never secure, and large-scale attackers could have found the break long before the public did.

MD5 was published in 1992 [99] with a claim of 2^{64} collision security. There were alarm bells from cryptanalysts, such as [52] ("it is anticipated that these techniques can be used to produce collisions for MD5"), but the claim was not publicly broken until 12 years later, when the results of [119] were announced. Followup work culminating in [112] exploited MD5 chosen-prefix collisions to efficiently forge certificates for arbitrary web sites. It was announced in 2012 that malware called "Flame" had been exploiting MD5 collisions since at least 2010; the analysis of [111] concluded that the Flame attackers had used an "entirely new and unknown" variant of [112] (meaning new from the public perspective), that the Flame design “required world-class cryptanalysis”, and that it was “not unreasonable to assume” that this cryptanalysis predated [112].

Think of a cryptographic proposal as a random variable, with some probability $p(M)$ of being publicly broken within M months. Assume for simplicity that these probabilities are independent across proposals. Let’s also optimistically assume that $p(\infty)$, the limit of $p(M)$ as M increases, captures all attacks that the attacker will find—the public will eventually find everything; and that the same probabilities apply specifically to submissions to competitions, rather than competitions tending to encourage weak submissions.

By collecting enough data, we can retrospectively estimate $p(M)$ for small values of M, and perhaps the curve would let us guess $p(\infty)$. For example, one can start by estimating $p(12) \approx 1/3$ given that 5 of the 15 AES submissions were publicly broken within 12 months, although obviously this selection of data should be replaced by much more data.

Consider a competition that chooses a random winner among the submissions not publicly broken within 36 months (assuming there is one). A submission has

- probability $p(36)$ of being publicly broken within 36 months,
- probability $p(\infty) - p(36)$ of being breakable but not publicly broken within 36 months, and
- probability $1 - p(\infty)$ of being secure.

The winner is thus breakable with probability $(p(\infty) - p(36))/(1 - p(36))$.

If, for example, data collection shows that there have been 1000 cryptographic proposals, with just 100 publicly broken within 36 months, and just 10 (like MD5) publicly broken between 36 months and 180 months, then $1 - p(36) = 0.9$ and

$$p(\infty) - p(36) \geq p(180) - p(36) = 0.01,$$

so the winner is breakable with probability at least 0.01/0.9. Does it make sense for cryptographers to worry about a user choosing a weak key with probability 2^{-64}, while not obviously worrying about each new cryptographic competition choosing a breakable winner with probability above 1%?

Now let’s partition the proposals into two types, 50% faster proposals and 50% slower proposals. Let’s define $p_1(M)$ as the conditional probability of a faster proposal being publicly broken within M months, and $p_2(M)$ as the conditional probability of a slower proposal being publicly broken within M months. By definition $p(M) = 0.5p_1(M) + 0.5p_2(M)$. Assume for simplicity that there are no further correlations between efficiency and brokenness.

Consider a competition that receives F faster submissions, receives infinitely many slower submissions, and chooses the *most efficient* submission not publicly broken within 36 months. The probability that all of the faster submissions are publicly broken within 36 months is $p_1(36)^F$, so the competition winner is breakable with probability

$$p_1(36)^F \frac{p_2(\infty) - p_2(36)}{1 - p_2(36)} + (1 - p_1(36)^F) \frac{p_1(\infty) - p_1(36)}{1 - p_1(36)}.$$

If $p_1(36)$ isn't too close to 1 and F isn't too small then $p_1(36)^F$ is close to 0, so the winner is breakable with probability close to $(p_1(\infty) - p_1(36))/(1 - p_1(36))$.

This probability could be even larger than $(p(\infty)-p(36))/(1-p(36))$, meaning that taking the most efficient unbroken submission is increasing risk compared to taking a random unbroken submission. It's easy to imagine reasons for faster proposals to be more likely to be broken than slower proposals—and more likely to be publicly broken after 36 months, as in the case of MD5. On the other hand, perhaps data collection will show that for some reason faster proposals are actually less risky overall. Even if they're more risky, perhaps this is outweighed by the benefit that they produce *for users taking the maximum security margin.* Similar comments apply when there are more than 2 different levels of efficiency.

Perhaps $p(M)$ and $p_1(M)$ should be stratified into $p(M, Y)$ and $p_1(M, Y)$, where Y is the year when a proposal was made. Optimists might hope that $p(M, Y)$ has been decreasing with Y. But many submissions to the most recent competitions have been broken (see, e.g., [**24**, PDF page 91]), showing that $p(M, Y)$ remains far above 0 for small M. Why shouldn't we think that $p(\infty, Y)$ is even larger than $p(36, Y)$, and that $p_1(\infty, Y)$ is even larger than $p_1(36, Y)$?

3.3. The overworked cryptanalyst. One way to argue that competitions reduce security risks is to argue that they focus the community's attention on finding all possible attacks. But does a competition provide enough time for this?

F-FCSR, one of the eight ciphers selected for the eSTREAM portfolio, was then shown in [**64**] to be very efficiently broken from 13 megabytes of output. As another example, the Rijndael designers and NIST claimed that Rijndael was "not vulnerable to timing attacks", but this was then disproven; as noted in Section 1, timing attacks continue to cause AES security problems today. These attacks work for any number of AES rounds, illustrating that security margins don't necessarily eliminate the security risks that remain after competitions.

If an attack takes years to develop, perhaps the reason is that it is the end of a long sequential chain of thoughts adding up to years of latency, but a simpler explanation is throughput. The world has a limited number of cryptographic experts capable of carrying out, and willing to carry out, public security analysis. This valuable time is divided across a huge number of proposals. This problem is more severe for competitions having more submissions—and for competitions having more complicated submissions. A competition might attract cryptanalyst time that would otherwise have been spent elsewhere, but the existence of a competition also *creates* submissions, cryptographic proposals that would not have existed otherwise, so it is far from clear that the amount of analysis per year per submission is larger than the amount of analysis per year per proposal outside competitions.

Perhaps having a critical mass of cryptanalysts focusing on one topic at the same time leads to breakthroughs that would not otherwise happen. Or perhaps the focus is wasting valuable cryptanalytic time on redundant parallel work, and cryptanalysts work more efficiently when they are roaming free through a larger field of targets.

competition	years
DES: the Data Encryption Standard	1974–1976
AES: the Advanced Encryption Standard	1998–2000
eSTREAM: the ECRYPT Stream Cipher Project	2005–2008
SHA-3: a Secure Hash Algorithm	2008–2012
CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness	2014–2019
NISTPQC: NIST Post-Quantum Cryptography Standardization Project	2017–?
NISTLWC: NIST Lightweight Cryptography Standardization Project	2019–?

competition	start	submissions	after $\mathbf{15 \pm 1}$
DES	M−13	M0: **1** → M26: **1**	
AES	M−17	M0: **15** → M14: **5** → M28: **1**	28 months
eSTREAM	M−5	M0: **34** → M11: **27** → M24: **16** → M36: **8**	12 months
SHA-3	M−21	M0: **51** → M9: **14** → M26: **5** → M48: **1**	39 months
CAESAR	M−14	M0: **56** → M16: **29** → M29: **15** → M48: **7** → M59: **6**	30 months
NISTPQC	M−20	M0: **69** → M13: **26** → M31: **15** → ?	
NISTLWC	M−30	M0: **56** → M6: **32** → M25: **10** → ?	

Fig. 3.4. Number of submissions remaining in each phase of various competitions. M0 is the calendar month when initial submissions were due. The first boldface number is the number of submissions allowed into the first phase. This is often smaller than the total number of submissions; e.g., DES disallowed most submissions, according to [49, Section 6]. Each subsequent M is the calendar month when submissions were announced for a subsequent phase, and the boldface number is the number of those submissions. For DES, AES, eSTREAM, SHA-3, and CAESAR, the final boldface number is the number selected for the portfolio as output of the competition, ending the competition. eSTREAM then updated its portfolio 5 months later to remove a broken portfolio member. NISTPQC and NISTLWC are ongoing and have not selected a portfolio yet. Fourth column is the number of months from 15 ± 1 candidates to the portfolio. Second column is when competition was announced. Range of years in the top table starts when initial submissions were due and ends when the portfolio was announced.

Competitions normally run through multiple phases, each phase narrowing the list of submissions. See Figure 3.4. A shorter and shorter list makes it more and more reasonable to believe that cryptanalysts are focusing on each remaining proposal more than what would have happened without a competition. AES, for example, narrowed the 15 initial submissions to 5 finalists, and presumably those were the top targets for security analysis at that point. On the other hand, final comments were due just 9 months later, and NIST announced the winner just 5 months after that. Concerns about not having enough time were expressed in [96, Section 2.1]:

> We believe that the effort spent on evaluating the security of the five AES finalists has been very limited, certainly compared to the 17 man-years spent by IBM on DES in the 1970s.

Sometimes competitions ask cryptanalysts to focus on particular submissions, while not necessarily excluding other submissions:

- eSTREAM's second phase selected 27 submissions, designating 10 as "focus" submissions: "These are designs that eSTREAM finds of particular interest. We particularly encourage more cryptanalysis and performance evaluation on these primitives." The others were "designs that eSTREAM wishes to move to the second phase of the eSTREAM project". (Two of the others, F-FCSR and Rabbit, ended up being in the eSTREAM portfolio.)
- NISTPQC's third phase selected 15 submissions, designating 7 as finalists: "NIST intends to select a small number of the finalists for standardization at the end of the third round. In addition, NIST expects to standardize a small number of the alternate candidates (most likely at a later date)."

Cryptanalysts can also decide on their own to focus on the fastest submissions, guessing that those are the easiest submissions to break, and the most likely to be selected if they are not broken. This might seem to be a successful strategy if the focus produces an attack—but why should we think that enough time has been spent analyzing the remaining submissions?

Section 2.9 considered ways to reduce the risk of the fastest function not being recognized as the fastest. One can analogously try to reduce the risk of the fastest function not being recognized as being breakable:

- Limit the complexity of the security goals for the competition, to better focus security-analysis work during the competition (although there is then a risk that these security goals are inadequate representatives of other security goals).
- Track expert assessments of the unexplored avenues of attack against each submission.
- Extend the competition time until the attack picture has settled down.

I suspect that collecting historical data will show that the security risks from later attack improvements have been quantitatively more severe, in probability and in impact, than the security risks arising from later performance improvements.

At the beginning of 2012, the SHA-3 competition was almost over, and the consensus of the cryptanalysts and designers I talked to was that it would be useful to have a new competition. Authenticated encryption was an obvious target—compared to stream ciphers and hash functions, authenticated ciphers are a step closer to the typical user's needs—and group discussions didn't identify any better targets. Interfaces even closer to the user's needs were identified (for example, secure sessions) but raised concerns of being too big a jump, needing too much new security analysis.

When I announced CAESAR at the beginning of 2013, I posted a "Timeline (tentative)" stretching five years into the future, with a submission deadline a year later and then four years of analysis ending with a portfolio. The actual schedule ended up lasting a year longer than the tentative schedule: submitters were asking for more time already from the outset (e.g., "the extra 2 months is a small price to pay if it increases the quality of submission pool (which I'm sure it will)"), cryptanalysts were asking for more time, etc. In the end CAESAR selected a portfolio of authenticated ciphers with exciting performance features and security features—see Section 2.5 for the list of ciphers—and *in the subsequent 27 months* nothing has publicly gone wrong with any of them.

3.5. Cryptographic risk management beyond timing. Public advances in attack algorithms generally begin with experts recognizing dangerous structure in the functions being attacked. Typically this structure can be described as an analogy to another broken function.

This doesn't mean that an expert recognizing dangerous structure is always able to find an attack. Often one cryptanalyst extends an attack on function F to an attack on function G, and then another cryptanalyst extends the attack on G to an attack on function H, where the first cryptanalyst already recognized the analogy between F and H and figured out the attack on G as one step from F towards H. Sometimes the first cryptanalyst already issues a warning regarding H, such as the MD5 alarm bells from [**52**] mentioned in Section 3.2.

A competition doesn't have to select the most efficient unbroken submission. It can try to reduce security risks by paying attention to extra information, namely the concerns that experts have regarding submissions. Factoring this information into decisions is complementary to giving cryptanalysts more time, the approach of Section 3.3.

I'm not saying that analogies always turn into attacks. It's easy to draw a chain of analogies from any cryptographic function to a broken function, so if analogies always turned into attacks then everything would be broken, which we hope isn't the case. There is also a procedural problem with simply downgrading any submission for which someone claims that there's a dangerous structure: this invites superficial claims from competing submissions.

I set a general policy of public evaluations for CAESAR:

> CAESAR selection decisions will be made on the basis of *published* analyses. If submitters disagree with published analyses then they are expected to promptly and *publicly* respond to those analyses. Any attempt to privately lobby the selection-committee members is contrary to the principles of public evaluation and should be expected to lead to disqualification.

Perhaps we can find clear rules reducing cryptographic risks, improving upon the baseline rule of eliminating publicly broken algorithms. Those rules can then be published, shown through analyses to be beneficial, and applied by everybody. But what happens when experts are spotting risks in a way that isn't captured by any known rules? Do we ignore those risks, or do we try to take them into account?

One answer is to put the experts onto the selection committee. I tried hard to fill the CAESAR selection committee with top symmetric cryptographers, people having the experience and judgment to see risks in advance. I promised, as part of inviting people to join the committee and as part of the public procedures for

CAESAR, that the committee would simply select algorithms and cite public analyses, rather than publishing its own analyses. Forcing the committee to publish analyses would have discouraged participation,[10] taking resources away from the core job of making judgment calls *beyond* published analyses.

Almost everyone I invited said yes. A few later ran out of time, but all of the following continued through the end (affiliations listed here are from when CAESAR began):

- Steve Babbage (Vodafone Group, UK)
- Alex Biryukov (University of Luxembourg, Luxembourg)
- Anne Canteaut (Inria Paris-Rocquencourt, France)
- Carlos Cid (Royal Holloway, University of London, UK)
- Joan Daemen (STMicroelectronics, Belgium)
- Orr Dunkelman (University of Haifa, Israel)
- Henri Gilbert (ANSSI, France)
- Tetsu Iwata (Nagoya University, Japan)
- Stefan Lucks (Bauhaus-Universität Weimar, Germany)
- Willi Meier (FHNW, Switzerland)
- Bart Preneel (COSIC, KU Leuven, Belgium)
- Vincent Rijmen (KU Leuven, Belgium)
- Matt Robshaw (Impinj, USA)
- Phillip Rogaway (University of California at Davis, USA)
- Greg Rose (Qualcomm, USA)
- Serge Vaudenay (EPFL, Switzerland)
- Hongjun Wu (Nanyang Technological University, Singapore)

I served on the committee as non-voting secretary, tracking the discussions and decisions and handling public communication.

I don't know how to prove that factoring in expert judgments is more reliable than simply taking the fastest unbroken algorithm. Maybe it isn't—or maybe there's a better approach. It would be beneficial for the cryptographic community to put more effort into analyzing and optimizing risk-management techniques.

3.6. The goal of limiting security. Performance pressures and limited time for security analysis are not the only sources of security risks in cryptographic competitions, as the history of DES illustrates.

According to a 1978 interview [75], DES product leader Walter Tuchman described DES as "the culmination of six years of research and development at IBM", a "three-pronged effort" involving his data-security-products group at IBM, the "mathematics department at IBM's Yorktown Heights research center", and "university consultants". IBM was then "ready to respond" when the National

[10] Consider an expert whose mental model says that unbroken algorithms A, B, C, D, and E have, respectively, chance 90%, 20%, 10%, 80%, and 60% of being secure, and consider a process that factors this information into decisions. The same expert sees ample evidence of the general public's limited understanding of what probabilities mean, and does not want to be subjected to complaints such as "E was broken after you said it was probably secure!".

Bureau of Standards (NBS, later renamed NIST) “issued its request for data encryption algorithm proposals”. Regarding accusations that IBM and NSA had “conspired”, Tuchman said “We developed the DES algorithm entirely within IBM using IBMers. The NSA did not dictate a single wire!”

In 1979, NSA director Bobby Inman gave a public speech [68] including the following comments: “First, let me set the record straight on some recent history. NSA has been accused of intervening in the development of the DES and of tampering with the standard so as to weaken it cryptographically. This allegation is totally false.” Inman continued with the following quote from a public 1978 Senate report [109]: “NSA did not tamper with the design of the algorithm in any way. IBM invented and designed the algorithm, made all pertinent decisions regarding it, and concurred that the agreed upon key size was more than adequate for all commercial applications for which the DES was intended.” This report was also mentioned in [75], which said that according to Tuchman the report had concluded “that there had been no collusion between IBM and the NSA”.

However, an internal NSA history book “American cryptology during the cold war” tells a story [71, pages 232–233] of much heavier NSA involvement in DES:

- NBS began to investigate encryption in 1968. NBS “went to NSA for help”.
- NSA’s “decision to get involved” with NBS on this was “hardly unanimous”. A “competent industry standard” could “spread into undesirable areas” such as “Third World government communications” and drugs and terrorism. On the other hand, “NSA had only recently discovered the large-scale Soviet pilfering of information from U.S. government and defense industry telephone communications. This argued the opposite case—that, as Frank Rowlett had contended since World War II, in the long run it was more important to secure one’s own communications than to exploit those of the enemy”.
- “Once that decision had been made, the debate turned to the issue of minimizing the damage. **Narrowing the encryption problem to a single, influential algorithm might drive out competitors, and that would reduce the field that NSA had to be concerned about. Could a public encryption standard be made secure enough to protect against everything but a massive brute force attack, but weak enough to still permit an attack of some nature using very sophisticated (and expensive) techniques?**” (Emphasis added. It is interesting to note the lack of any consideration of the possibility that any cryptosystem weak enough to be breakable by NSA would also be breakable by the Soviets.)
- Back to NBS: It “was decided” that NBS would “use the *Federal Register* to solicit the commercial sector for an encryption algorithm”. NSA would “evaluate the quality, and if nothing acceptable appeared, would devise one itself”.
- The response to NBS’s 1973 call for proposals “was disappointing, so NSA began working on its own algorithm”. NSA then “discovered that Walter Tuchman of IBM was working on a modification to Lucifer for general use. **NSA gave Tuchman a clearance and brought him in to work jointly with the Agency on his Lucifer modification**”. (Emphasis added.)

- Regarding the goal of making sure DES was “strong enough” but also “weak enough”: “NSA worked closely with IBM to strengthen the algorithm against all except brute force attacks and to strengthen substitution tables, called S-boxes. Conversely, NSA tried to convince IBM to reduce the length of the key from 64 to 48 bits. Ultimately, they compromised on a 56-bit key.” (For comparison, the Senate report had stated that “NSA convinced IBM that a reduced key size was sufficient” and that NSA had “indirectly assisted in the development of the S box structures”.)
- “The relationship between NSA and NBS was very close. NSA scientists working the problem crossed back and forth between the two agencies, and **NSA unquestionably exercised an influential role in the algorithm**.” (Emphasis added.)

The relevant portions of this book became public because, starting in 2006, the non-profit National Security Archive (abbreviated “The Archive”, not “NSA”) filed a series of declassification requests and appeals [88] regarding the book. This forced portions of the book to be released in 2008, and further portions to be released in 2013—officially documenting, for example, NSA surveillance of Martin Luther King, Jr., journalist Tom Wicker, and senator Frank Church. There were also some intermediate releases from the book in response to a FOIA request filed by John Young in 2009; see [126].

To summarize, NSA worked with NBS on the DES competition before the competition was announced, and worked jointly with IBM on the design of DES before the final design was submitted to the competition. NSA’s actual goals for the competition—goals that it acted upon, with considerable success—included (1) making sure that DES was “weak enough” to be breakable by NSA and (2) having DES be “influential” enough to “drive out competitors”. The first goal directly threatens security, and the second goal extends the security damage.

3.7. The difficulty of recognizing attackers. An obvious response to the type of attack described in Section 3.6 is to set up competition procedures that exclude NSA—and other known attackers—from participation. However, NSA can secretly hire consultants to participate in the competitions and to try to weaken security in the same way that NSA would have. These consultants can deny NSA involvement, the same way Tuchman did.

The core problem is that it is not easy to recognize attackers. It is instructive to look back at the extent to which the cryptographic community has failed to recognize NSA as an attacker, never mind the harder problem of recognizing others working with NSA as attackers.

After differential cryptanalysis was published but was shown to have less impact on DES than on many DES variants, Coppersmith revealed [43] that the DES design team had already known about differential cryptanalysis and had designed DES accordingly. This differential-cryptanalysis story contributed to a pervasive “good guys” narrative claiming that NSA had *strengthened* IBM’s DES design. Here are two examples of this narrative appearing in response to concerns regarding NSA influence:

- NIST's standard elliptic curves were designed by NSA, and were claimed to be "verifiably random". Scott [108] pointed out that if NSA knew a weakness in one curve in a million then the claimed "verifiable randomness" would not have stopped NSA from selecting a weak curve; see also [26], [25], and [27]. For a "good guys" response, see [61], which, in reply to [26], stated the following: "Flipside: What if NIST/NSA know a weakness in 1/10000000 curves? NIST searches space for curves that *aren't* vulnerable." (The same author later stated [62] that this comment was from when he was "younger and more naive".)
- NSA budget documents leaked in September 2013 listed 0.25 billion dollars per year for a "SIGINT Enabling Project" that "actively engages the U.S. and foreign IT industries to covertly influence and/or overtly leverage their commercial products' designs" to make them "exploitable" [94], including a goal to "influence policies, standards and specifications for commercial public key technologies". This is what one would expect from an agency whose primary mission has always been signals intelligence; and it is consistent with NSA's early-1970s goal, quoted above, of ensuring that DES was "weak enough to still permit an attack of some nature". For a "good guys" response, see [35], which portrays the "SIGINT Enabling Project" as something new: [35] has subtitle "Leaked documents say that the NSA has compromised encryption specs. It wasn't always this way"; claims that NSA's "secretive work on DES" had "made the algorithm better"; and asks if there was a "change in mission".

See also [69], [127], and [123].

The disclosures in 2013 did not stop NSA from participating in processes to select cryptographic algorithms. See, e.g., [13], describing NSA's efforts between 2014 and 2018 to convince ISO to standardize Simon and Speck. One can only guess how many more algorithm-selection processes NSA was influencing through proxies in the meantime.

CAESAR began before those disclosures, but I was already well aware of NSA's role as an attacker; see, e.g., [17]. I hoped that having as much as possible done in public would, beyond its basic advantages in correcting errors, also stop NSA and other attackers from sabotaging the process. Obviously attackers could still submit algorithms, but one of the required sections of each submission was the following:

> **Design rationale:** An explanation of the choices made in the cipher design. This section is expected to include an analysis of how a weakness could be hidden in the cipher. This section must include the following statement: "The designer/designers have not hidden any weaknesses in this cipher."

An attacker can easily lie about the existence of weaknesses, but being forced to explain the choices made in the design gives the community and the committee a chance to catch inadequate explanations.

Would an attacker be able to sneak a weak algorithm through a committee full of experts? Would it be able to sneak a weak algorithm through a competition that simply takes the fastest unbroken algorithm? These are interesting questions to analyze.

3.8. The goal of producing publications. I'll close this paper by describing one more incentive that creates security risks in cryptographic competitions, an incentive that also explains many phenomena described earlier in this paper.

As academic cryptographers, we're paid primarily to produce publications, specifically papers. Most—although not all—deployed systems come from papers written by academic cryptographers, after passing through a long supply chain involving people paid to produce cryptographic standards, and people paid to produce cryptographic libraries, and so on.

When a cryptographic system fails, we blame *that system* for failing, as noted in Section 1. We then use the failure as motivation to write papers proposing and analyzing new systems. If the broken system is important enough then this also means new versions of standards, new versions of libraries, etc.

We all have a perverse incentive to stay in this situation, collectively creating a neverending series of cryptosystems failing in supposedly new and exciting ways, so that we can continue writing papers designing and analyzing the next systems. Papers and grant proposals on improved attacks and improved cryptosystems and security proofs habitually explain their importance by citing recent failures. If we instead give the users "boring crypto" [19]—"crypto that simply works, solidly resists attacks, never needs any upgrades"—then will our readers and funding agencies still be interested in our subsequent papers? Perhaps, but do we really want to take this chance?

If we have boring block ciphers then as a community we could move on to, say, stream ciphers, and if we have boring stream ciphers then we could move on to authenticated ciphers, and if we have boring authenticated ciphers then we could move on to secure sessions. But won't the users say at some point that they have exactly the right cryptographic operations and don't need further input from us? It's safer for us if our core cryptography keeps failing.

The cryptographic community as a whole systematically flunks Taleb's "skin in the game" requirement [113] for risk management. As cryptographers, how often do we think that the damage caused by cryptographic failures will make *us* suffer? The designers of a system don't expect it to be broken in the first place. If it *is* broken then hey, look, the designers have another citation, and now we can all write followup papers. It's against community standards to blame the designers rather than blaming the broken system. At worst there's a brief moment of embarrassment if the attack was "too easy".

We put *some* sort of requirements on cryptosystems to control the size of the literature and maintain the prestige of publications—e.g., any new block cipher must identify some performance metric where the cipher outperforms previous ciphers, and must include certain types of cryptanalysis—but we have little incentive to match these requirements to what the users want. What the users want, most importantly, is for us to be super-careful about security, but we have personal and community incentives against this. Being more careful than whatever is required for a publication is taking time away from writing more papers, and as a community we want a sufficiently steady stream of broken cryptosystems as continued fuel for the fire.

Imagine a competition requiring every cipher to have twice as many rounds as it seems to need. This would make typical attack improvements less scary, and would eliminate most—although not all—cipher breaks. This would make papers harder to publish. The community thus has an incentive to argue *against* such a requirement, claiming that it's overkill and claiming that a 2× slowdown is a problem. To support such arguments, we generate even more papers, such as performance-analysis papers saying that Serpent is slower than Rijndael.

More broadly, performance seems to be the most powerful weapon we have in the fight against ideas for reducing security risks. Performance constantly drives us towards the edge of disaster, and that's what we want. The edge is interesting. The edge produces papers. This also produces an incentive for us to continually claim that performance matters, and an incentive for us to avoid investigating the extent to which this is true. See Section 2.

A traditional report on the CAESAR competition would say that it produced many papers, advancing researchers' understanding of security and performance, building a foundation for the next generation of papers on symmetric cryptology. All of these things are true. DES was already a success in these metrics, and subsequent competitions have been even more successful. The challenge for the community is to figure out whether we can maintain success in what we're paid to do *without* a neverending series of security failures.

References

[1] — (no editor), *Redacted transcript of hearing of House International Relations Committee on 21 July 1997* (1997). URL: https://cryptome.org/jya/hir-hear.htm. Citations in this document: §1.

[2] — (no editor), *International conference on field programmable logic and applications, FPL 2010, August 31 2010–September 2, 2010, Milano, Italy*, IEEE Computer Society, 2010. ISBN 978-0-7695-4179-2. See [57].

[3] — (no editor), *27th annual network and distributed system security symposium, NDSS 2020, San Diego, California, USA, February 23–26, 2020*, The Internet Society, 2020. ISBN 1-891562-61-4. See [110].

[4] — (no editor), *2020 IEEE Symposium on Security and Privacy (S&P 2020), 17–21 May 2020, San Francisco, California, USA*, Institute of Electrical and Electronics Engineers, 2020. ISBN 978-1-7281-3497-0. See [42].

[5] — (no editor), *Proceedings of the 29th USENIX security symposium, August 12–14, 2020*, USENIX, 2020. See [81].

[6] Akamai, *Akamai online retail performance report: Milliseconds are critical* (2017). URL: https://www.akamai.com/uk/en/about/news/press/2017-press/akamai-releases-spring-2017-state-of-online-retail-performance-report.jsp. Citations in this document: §2.2.

[7] Akamai, *The state of online retail performance* (2017). URL: https://www.akamai.com/us/en/multimedia/documents/report/akamai-state-of-online-retail-performance-spring-2017.pdf. Citations in this document: §2.2, §2.3, §2.2, §2.2, §2.2.

[8] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, *Status report on the first round of the NIST Post-Quantum Cryptography Standardization Process*, NISTIR 8240 (2019). URL: https://csrc.nist.gov/publications/detail/nistir/8240/final. Citations in this document: §1.1.

[9] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, *Status report on the second round of the NIST Post-Quantum Cryptography Standardization Process*, NISTIR 8309. URL: https://csrc.nist.gov/publications/detail/nistir/8309/final. Citations in this document: §1.1, §2.5.

[10] Mark Allman, Sally Floyd, Craig Partridge, *Increasing TCP's initial window* (1998); see also newer version [**11**]. URL: https://www.rfc-editor.org/rfc/rfc2414. Citations in this document: §2.4.

[11] Mark Allman, Sally Floyd, Craig Partridge, *Increasing TCP's initial window* (2002); see also older version [**10**]. URL: https://www.rfc-editor.org/rfc/rfc3390. Citations in this document: §2.4.

[12] ARM, *ARM extends 28nm IP leadership with latest UMC 28HPC POPs* (2016). URL: https://www.arm.com/company/news/2016/02/arm-extends-28nm-ip-leadership-with-latest-umc-28hpc-pops. Citations in this document: §2.1.

[13] Tomer Ashur, Atul Luykx, *An account of the ISO/IEC standardization of the Simon and Speck Block Cipher Families* (2018). URL: https://www.esat.kuleuven.be/cosic/publications/article-2957.pdf. Citations in this document: §3.7.

[14] Steve Babbage, Christophe De Cannière, Anne Canteaut, Carlos Cid, Henri Gilbert, Thomas Johansson, Matthew Parker, Bart Preneel, Vincent Rijmen, Matthew Robshaw, *The eSTREAM portfolio* (2008). URL: https://www.ecrypt.eu.org/stream/portfolio.pdf. Citations in this document: §2.5.

[15] Sonia Belaïd, Tim Güneysu (editors), *Smart card research and advanced applications—18th international conference, CARDIS 2019, Prague, Czech Republic, November 11–13, 2019, revised selected papers*, Lecture Notes in Computer Science, 11833, Springer, 2019. ISBN 978-3-030-42067-3. See [53].

[16] Daniel J. Bernstein, *Cache-timing attacks on AES* (2005). URL: https://cr.yp.to/papers.html#cachetiming. Citations in this document: §1.

[17] Daniel J. Bernstein, *Cryptography for the paranoid*, slides (2012). URL: https://cr.yp.to/talks.html#2012.09.24. Citations in this document: §3.7.

[18] Daniel J. Bernstein, *CAESAR call for submissions* (2014). URL: https://competitions.cr.yp.to/caesar-call.html. Citations in this document: §2.5.

[19] Daniel J. Bernstein, *Boring crypto*, slides (2015). URL: https://cr.yp.to/talks.html#2015.10.05. Citations in this document: §3.8.

[20] Daniel J. Bernstein, *CAESAR use cases* (2016). URL: https://groups.google.com/g/crypto-competitions/c/DLv193SPSDc/m/4CeHPvIoBgAJ. Citations in this document: §2.5.

[21] Daniel J. Bernstein (editor), *Challenges in authenticated encryption*, ECRYPT-CSA D1.1, revision 1.05 (2017). URL: https://chae.cr.yp.to/chae-20170301.pdf. Citations in this document: §2.5, §A.

[22] Daniel J. Bernstein, *Visualizing size-security tradeoffs for lattice-based encryption*, Second PQC Standardization Conference (2019). URL: https://cr.yp.to/papers.html#paretoviz. Citations in this document: §3.1.

[23] Daniel J. Bernstein, *Comparing proofs of security for lattice-based encryption*, Second PQC Standardization Conference (2019). URL: https://cr.yp.to/papers.html#latticeproofs. Citations in this document: §1.1.

[24] Daniel J. Bernstein, *Post-quantum cryptography* (2020). URL: https://cr.yp.to/talks.html#2020.01.30. Citations in this document: §3.2.

[25] Daniel J. Bernstein, Tung Chou, Chitchanok Chuengsatiansup, Andreas Hülsing, Eran Lambooij, Tanja Lange, Ruben Niederhagen, Christine van Vredendaal, *How to manipulate curve standards: a white paper for the black hat*, in SSR 2015 (2015). URL: https://bada55.cr.yp.to/. Citations in this document: §1.1, §3.7.

[26] Daniel J. Bernstein, Tanja Lange, *Security dangers of the NIST curves* (2013). URL: https://cr.yp.to/talks.html#2013.09.16. Citations in this document: §3.7, §3.7.

[27] Daniel J. Bernstein, Tanja Lange, *Failures in NIST's ECC standards* (2016). URL: https://cr.yp.to/papers.html#nistecc. Citations in this document: §3.7.

[28] Daniel J. Bernstein, Tanja Lange (editors), *eBACS: ECRYPT Benchmarking of Cryptographic Systems*, accessed 22 December 2020 (2020). URL: https://bench.cr.yp.to. Citations in this document: §2.10.

[29] Daniel J. Bernstein, Tanja Lange, *McTiny: fast high-confidence post-quantum key erasure for tiny network servers*, in USENIX 2020 [**37**] (2020), 1731–1748. URL: https://cr.yp.to/papers.html#mctiny. Citations in this document: §2.1.

[30] Daniel J. Bernstein, Tanja Lange, Ruben Niederhagen, *Dual EC: a standardized back door*, in [**105**] (2015), 256–281. URL: https://eprint.iacr.org/2015/767. Citations in this document: §1.1.

[31] Karthikeyan Bhargavan, Gaëtan Leurent, *On the practical (in-)security of 64-bit block ciphers: collision attacks on HTTP over TLS and OpenVPN*, in CCS 2016 [**120**] (2016), 456–467. Citations in this document: §1.

[32] Eli Biham, *How to forge DES-encrypted messages in* 2^{28} *steps*, Technion Computer Science Department Technical Report CS0884 (1996). URL: https://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/1996/CS/CS0884.pdf. Citations in this document: §1.

[33] Eli Biham, *A note on comparing the AES candidates*, in Second AES Candidate Conference (1999), 85–92. URL: https://cs.technion.ac.il/~biham/publications.html. Citations in this document: §2.6, §2.7.

[34] Dennis K. Branstad (editor), *Computer security and the Data Encryption Standard: proceedings of the conference on computer security and the Data Encryption Standard held at the National Bureau of Standards in Gaithersburg, Maryland on February 15, 1977*, NBS Special Publication 500-27, 1977. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nbsspecialpublication500-27.pdf. See [49].

[35] Peter Bright, *The NSA's work to make crypto worse and better* (2013). URL: https://arstechnica.com/information-technology/2013/09/the-nsas-work-to-make-crypto-worse-and-better/. Citations in this document: §3.7, §3.7.

[36] Christophe De Cannière, *eSTREAM Optimized Code HOWTO* (2005). URL: https://www.ecrypt.eu.org/stream/perf/. Citations in this document: §2.10, §2.10.

[37] Srdjan Capkun, Franziska Roesner (editors), *Proceedings of the 29th USENIX Security Symposium*, USENIX Association, 2020. ISBN 978-1-939133-17-5. See [29].

[38] Matthew R. Carter, Raghurama R. Velagala, John Pham, Jens-Peter Kaps, *eXtended eXternal benchmarking eXtension (XXBX)*, demo at IEEE Hardware Oriented Security and Trust 2018 (2018). URL: http://www.hostsymposium.org/host2018/hwdemo/HOST_2017_hwdemo_23.pdf. Citations in this document: §2.10.

[39] CDN Planet, *Initcwnd settings of major CDN providers* (2017). URL: https://www.cdnplanet.com/blog/initcwnd-settings-major-cdn-providers/. Citations in this document: §2.4, §2.4.

[40] Shu-jen Chang, Ray Perlner, William E. Burr, Meltem Sönmez Turan, John M. Kelsey, Souradyuti Paul, Lawrence E. Bassham, *Third-round report of the SHA-3 cryptographic hash algorithm competition*, NISTIR 7896 (2012). URL: https://csrc.nist.gov/publications/detail/nistir/7896/final. Citations in this document: §1.1, §2.5, §2.5, §2.10.

[41] Jerry Chu, Nandita Dukkipati, Yuchung Cheng, Matt Mathis, *Increasing TCP's initial window* (2013). URL: https://tools.ietf.org/html/rfc6928. Citations in this document: §2.4.

[42] Shaanan Cohney, Andrew Kwong, Shahar Paz, Daniel Genkin, Nadia Heninger, Eyal Ronen, Yuval Yarom, *Pseudorandom black swans: cache attacks on CTR_DRBG*, in S&P 2020 [4] (2020), 1241–1258. URL: https://eprint.iacr.org/2019/996. Citations in this document: §1, §A.

[43] Don Coppersmith, *The Data Encryption Standard (DES) and its strength against attacks*, IBM Journal of Research and Development **38** (1994), 243–250. URL: https://simson.net/ref/1994/coppersmith94.pdf. Citations in this document: §3.7.

[44] Ronald Cramer (editor), *Advances in cryptology—EUROCRYPT 2005, 24th annual international conference on the theory and applications of cryptographic techniques, Aarhus, Denmark, May 22–26, 2005, proceedings*, Lecture Notes in Computer Science, 3494, Springer, 2005. ISBN 3-540-25910-4. See [119].

[45] Paul Crowley, Eric Biggers, *Adiantum: length-preserving encryption for entry-level processors*, IACR Transactions on Symmetric Cryptology **2018** (2018), 39–61. URL: https://eprint.iacr.org/2018/720. Citations in this document: §2.1.

[46] Paul Crowley, Eric Biggers, *Introducing Adiantum: encryption for the next billion users* (2019). URL: https://security.googleblog.com/2019/02/introducing-adiantum-encryption-for.html. Citations in this document: §2.1.

[47] Cryptographic Technology Group, *NIST cryptographic standards and guidelines development process*, NISTIR 7977 (2016). URL: https://csrc.nist.gov/publications/detail/nistir/7977/final. Citations in this document: §1.1.

[48] Joan Daemen, Vincent Rijmen, *Resistance against implementation attacks: a comparative study of the AES proposals* (1999). URL: https://web.archive.org/web/20000816072451/http://csrc.nist.gov/encryption/aes/round1/conf2/papers/daemen.pdf. Citations in this document: §1.

[49] Ruth M. Davis, *The Data Encryption Standard in perspective*, in [**34**] (1977), 4–13. Citations in this document: §1, §1.1, §5, §3.4, §3.4.

[50] Whitfield Diffie, Martin E. Hellman, *Exhaustive cryptanalysis of the NBS Data Encryption Standard*, Computer **10** (1977), 74–84. URL: https://ee.stanford.edu/~hellman/publications/27.pdf. Citations in this document: §1, §2.5, §A.

[51] Dumitru-Daniel Dinu, Alex Biryukov, Johann Groszschädl, Dmitry Khovratovich, Yann Le Corre, Léo Perrin, *FELICS—fair evaluation of lightweight cryptographic systems*, NIST Workshop on Lightweight Cryptography 2015 (2015). URL: https://hdl.handle.net/10993/25967. Citations in this document: §2.10.

[52] Hans Dobbertin, Antoon Bosselaers, Bart Preneel, *RIPEMD-160: a strengthened version of RIPEMD*, in FSE 1996 [**59**] (1996), 71–82. Citations in this document: §3.2, §3.5.

[53] Luan Cardoso Dos Santos, Johann Groszschädl, Alex Biryukov, *FELICS-AEAD: benchmarking of lightweight authenticated encryption algorithms*, in CARDIS 2019 [**15**] (2019), 216–233. URL: https://hdl.handle.net/10993/41537. Citations in this document: §2.10.

[54] Orr Dunkelman, Léo Perrin, *Adapting rigidity to symmetric cryptography: towards "unswerving" designs*, in SSR 2019 (2019). URL: https://eprint.iacr.org/2019/1187. Citations in this document: §1.1.

[55] ECRYPT, *Call for stream cipher primitives* (2005). URL: https://www.ecrypt.eu.org/stream/call/. Citations in this document: §2.5.

[56] Electronic Frontier Foundation, *Cracking DES: secrets of encryption research, wiretap politics & chip design*, O'Reilly, 1998. ISBN 978-1565925205. Citations in this document: §1.

[57] Kris Gaj, Jens-Peter Kaps, Venkata Amirineni, Marcin Rogawski, Ekawat Homsirikamol, Benjamin Y. Brewster, *ATHENa—Automated Tool for Hardware EvaluatioN: toward fair and comprehensive benchmarking of cryptographic hardware using FPGAs*, in FPL 2010 [**2**] (2010), 414–421. Citations in this document: §2.10.

[58] Brian Gladman, *Serpent S boxes as Boolean functions* (2000). URL: https://web.archive.org/web/20001118002700/http://www.btinternet.com/~brian.gladman/cryptography_technology/serpent/index.html. Citations in this document: §2.9, §2.9, §2.9, §2.9.

[59] Dieter Gollmann (editor), *Fast software encryption, third international workshop, Cambridge, UK, February 21–23, 1996, proceedings*, Lecture Notes in Computer Science, 1039, Springer, 1996. ISBN 3-540-60865-6. See [52].

[60] Google, *Addressing performance regressions* (2021). URL: https://web.archive.org/web/20211226054614/https://chromium.googlesource.com/chromium/src/+/refs/heads/main/docs/speed/addressing_performance_regressions.md. Citations in this document: §2.4.

[61] Matthew D. Green, *"Flipside: What if NIST/NSA know a weakness in 1/10000000 curves? NIST searches space for curves that *aren't* vulnerable."*, tweet (2013). URL: https://archive.today/HyWGr. Citations in this document: §3.7.

[62] Matthew D. Green, *"Discussion with @hashbreaker from when I was younger and more naive. #nist #ecc"*, tweet (2013). URL: https://archive.is/59Hiz. Citations in this document: §3.7.

[63] Shai Halevi (editor), *Advances in cryptology—CRYPTO 2009, 29th annual international cryptology conference, Santa Barbara, CA, USA, August 16–20, 2009, proceedings*, Lecture notes in Computer Science, 5677, Springer, 2009. See [112].

[64] Martin Hell, Thomas Johansson, *Breaking the F-FCSR-H stream cipher in real time*, in Asiacrypt 2008 [**95**] (2008), 557–569. Citations in this document: §3.3.

[65] Martin E. Hellman, *A cryptanalytic time-memory tradeoff*, IEEE Transactions on Information Theory **26** (1980), 401–406. URL: https://ee.stanford.edu/~hellman/publications/36.pdf. Citations in this document: §1.

[66] Martin E. Hellman, Whitfield Diffie, Paul Baran, Dennis Branstad, Douglas L. Hogan, Arthur J. Levenson, *DES (Data Encryption Standard) review at Stanford University* (1976). URL: `https://web.archive.org/web/20170420171412/www.toad.com/des-stanford-meeting.html`. Citations in this document: §1.

[67] HTTP Archive, *Report: state of the web* (2020). URL: `https://httparchive.org/reports/state-of-the-web`. Citations in this document: §2.4.

[68] Bobby R. Inman, *The NSA perspective on telecommunications protection in the nongovernmental sector* (1979). URL: `https://cryptome.org/nsa-inman-1979.pdf`. Citations in this document: §1, §3.6.

[69] William Jackson, *NSA reveals its secret: No backdoor in encryption standard* (2011). URL: `https://gcn.com/articles/2011/02/16/rsa-11-nsa--no-des-backdoor.aspx`. Citations in this document: §3.7.

[70] Van Jacobson, *Congestion avoidance and control*, ACM SIGCOMM Computer Communication Review **18** (1988), 314–329. Citations in this document: §2.4.

[71] Thomas R. Johnson, *American cryptology during the cold war, 1945–1989, book III: retrenchment and reform, 1972–1980*, 1998. URL: `https://archive.org/details/cold_war_iii-nsa`. Citations in this document: §3.6.

[72] Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe, Ko Stoffelen, `pqm4`*: testing and benchmarking NIST PQC on ARM Cortex-M4* (2019). URL: `https://eprint.iacr.org/2019/844`. Citations in this document: §2.10.

[73] Jens-Peter Kaps, William Diehl, Michael Tempelmeier, Farnoud Farahmand, Ekawat Homsirikamol, Kris Gaj, *A comprehensive framework for fair and efficient benchmarking of hardware implementations of lightweight cryptography* (2019). URL: `https://eprint.iacr.org/2019/1273`. Citations in this document: §2.10.

[74] Richard F. Kayser, *Announcing request for candidate algorithm nominations for a new cryptographic hash algorithm (SHA3) family*, Federal Register **72** (2007), 62212–62220. URL: `https://www.govinfo.gov/content/pkg/FR-2007-11-02/pdf/E7-21581.pdf`. Citations in this document: §2.5.

[75] Paul Kinnucan, *Data encryption gurus: Tuchman and Meyer*, Cryptologia **2** (1978), 371–381. Citations in this document: §3.6, §3.6.

[76] Donald Knuth, *Structured programming with go to statements*, Computing Surveys **6** (1974), 261–301. Citations in this document: §2.7.

[77] Samuel Kramer, *Announcing development of a Federal Information Processing Standard for Advanced Encryption Standard*, Federal Register **62** (1996), 93–94. URL: `https://www.govinfo.gov/content/pkg/FR-1997-01-02/pdf/96-32494.pdf`. Citations in this document: §2.5.

[78] Adam Langley, *Maybe skip SHA-3* (2017). URL: `https://www.imperialviolet.org/2017/05/31/skipsha3.html`. Citations in this document: §1.

[79] David P. Leech, Michael W. Chinworth, *The economic impacts of NIST's data encryption standard (DES) program* (2001). URL: `https://csrc.nist.gov/publications/detail/white-paper/2001/10/01/the-economic-impacts-of-nist-des-program/final`. Citations in this document: §5, §5, §5.

[80] David P. Leech, Stacey Ferris, John T. Scott, *The economic impacts of the advanced encryption standard, 1996–2017* (2018). URL: `https://csrc.nist.gov/publications/detail/white-paper/2018/09/07/economic-impacts-of-the-advanced-encryption-standard-1996-2017/final`. Citations in this document: §2.9.

[81] Gaëtan Leurent, Thomas Peyrin, *SHA-1 is a shambles: first chosen-prefix collision on SHA-1 and application to the PGP web of trust*, in USENIX 2020

[5] (2020), 1839–1856. URL: https://eprint.iacr.org/2020/014. Citations in this document: §1.

[82] Stefan Mangard, François-Xavier Standaert (editors), *Cryptographic hardware and embedded systems, CHES 2010, 12th international workshop, Santa Barbara, CA, USA, August 17–20, 2010, proceedings*, Lecture Notes in Computer Science, 6225, Springer, 2010. ISBN 978-3-642-15030-2. See [121].

[83] David McGrew, *Impossible plaintext cryptanalysis and probable-plaintext collision attacks of 64-bit block cipher modes* (2012). URL: https://eprint.iacr.org/2012/623. Citations in this document: §1.

[84] Paul Meissner (editor), *Report of the workshop on estimation of significant advances in computer technology*, NBSIR 76-1189 (1976). URL: https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nbsir76-1189.pdf. Citations in this document: §2.5.

[85] Kaushik Nath, Palash Sarkar, *Efficient 4-way vectorizations of the Montgomery ladder* (2020). URL: https://eprint.iacr.org/2020/378. Citations in this document: §2.1.

[86] National Institute of Standards and Technology, *NIST's efficiency testing for Round1 AES candidates* (1999). URL: https://web.archive.org/web/20000816072005/https://csrc.nist.gov/encryption/aes/round1/conf2/NIST-efficiency-testing.pdf. Citations in this document: §2.6, §2.7.

[87] National Institute of Standards and Technology, *Submission requirements and evaluation criteria for the lightweight cryptography standardization process* (2018). URL: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf. Citations in this document: §2.5.

[88] National Security Archive, *"Disreputable if not outright illegal": the National Security Agency versus Martin Luther King, Muhammad Ali, Art Buchwald, Frank Church, et al.* (2013). URL: https://nsarchive2.gwu.edu/NSAEBB/NSAEBB441/. Citations in this document: §3.6.

[89] James Nechvatal, Elaine Barker, Lawrence Bassham, William Burr, Morris Dworkin, James Foti, Edward Roback, *Report on the development of the Advanced Encryption Standard (AES)*, Journal of Research of the National Institute of Standards and Technology **106** (2001). URL: https://nvlpubs.nist.gov/nistpubs/jres/106/3/j63nec.pdf. Citations in this document: §1, §1.1, §2.5, §2.5, §2.9, §2.9, §2.10, §A.

[90] James Nechvatal, Elaine Barker, Donna Dodson, Morris Dworkin, James Foti, Edward Roback, *Status report on the first round of the development of the Advanced Encryption Standard*, Journal of Research of the National Institute of Standards and Technology **104** (1999). URL: https://nvlpubs.nist.gov/nistpubs/jres/104/5/j45nec.pdf. Citations in this document: §1.1.

[91] OpenDNS Team, *OpenDNS adopts DNSCurve* (2010). URL: https://umbrella.cisco.com/blog/opendns-dnscurve. Citations in this document: §2.1.

[92] Dag Arne Osvik, *Speeding up Serpent*, in Third AES Candidate Conference (2000), 317–329. URL: https://www.ii.uib.no/~osvik/pub/aes3.pdf. Citations in this document: §2.6, §2.9, §2.9, §2.9.

[93] Elisabeth Paté-Cornell, Louis Anthony Cox Jr., *Improving risk management: from lame excuses to principled practice*, Risk Analysis **34** (2014), 1228–1239. Citations in this document: §1.1.

[94] Nicole Perlroth, Jeff Larson, Scott Shane, *N.S.A. able to foil basic safeguards of privacy on web* (2013). URL: https://www.nytimes.com/2013/09/06/us/nsa-foils-much-internet-encryption.html. Citations in this document: §3.7.

[95] Josef Pieprzyk (editor), *Advances in cryptology—ASIACRYPT 2008, 14th international conference on the theory and application of cryptology and information security, Melbourne, Australia, December 7–11, 2008. proceedings*, Lecture Notes in Computer Science, 5350, Springer, 2008. ISBN 978-3-540-89254-0. See [64].

[96] Bart Preneel, Antoon Bosselaers, Vincent Rijmen, Bart Van Rompay, Louis Granboulan, Jacques Stern, Sean Murphy, Markus Dichtl, Pascale Serf, Eli Biham, Orr Dunkelman, Vladimir Furman, François Koeune, Gilles Piret, Jean-Jacques Quisquater, Lars Knudsen, Håvard Raddum, *Comments by the NESSIE project on the AES finalists* (2000). URL: https://www.cosic.esat.kuleuven.be/nessie/deliverables/D4_NessieAESInput.pdf. Citations in this document: §1.1, §3.3.

[97] Bart Preneel, Bart Van Rompay, Siddika Berna Örs, Alex Biryukov, Louis Granboulan, Emmanuelle Dottax, Markus Dichtl, Marcus Schafheutle, Pascale Serf, Stefan Pyka, Eli Biham, Elad Barkan, Orr Dunkelman, J. Stolin, Matthieu Ciet, Jean-Jacques Quisquater, Francisco Sica, Håvard Raddum, Matthew Parker, *Performance of optimized implementations of the NESSIE primitives* (2003). URL: https://www.cosic.esat.kuleuven.be/nessie/deliverables/D21-v2.pdf. Citations in this document: §2.10.

[98] Andrew Regenscheid, Ray Perlner, Shu-jen Chang, John Kelsey, Mridul Nandi, Souradyuti Paul, *Status report on the first round of the SHA-3 cryptographic hash algorithm competition*, NISTIR 7620 (2009). URL: https://csrc.nist.gov/publications/detail/nistir/7620/final. Citations in this document: §1.1.

[99] Ronald L. Rivest, *The MD5 message-digest algorithm*, RFC 1321 (1992). URL: https://tools.ietf.org/html/rfc1321. Citations in this document: §3.2.

[100] Matthew Robshaw, *The eSTREAM project*, in [**101**] (2008), 1–6. Citations in this document: §1.1.

[101] Matthew Robshaw, Olivier Billet (editors), *New stream cipher designs: the eSTREAM finalists*, Lecture Notes in Computer Science, 4986, Springer, 2008. ISBN 978-3-540-68350-6. See [100].

[102] Root Server Operators, *Events of 2015-11-30* (2015). URL: https://root-servers.org/media/news/events-of-20151130.txt. Citations in this document: §2.1.

[103] Root Server Operators, *Threat mitigation for the root server system* (2019). URL: https://root-servers.org/media/news/Threat_Mitigation_For_the_Root_Server_System.pdf. Citations in this document: §2.1.

[104] Root Server Operators, *Statement on DNS encryption* (2021). URL: https://root-servers.org/media/news/Statement_on_DNS_Encryption.pdf. Citations in this document: §2.1.

[105] Peter Y. A. Ryan, David Naccache, Jean-Jacques Quisquater (editors), *The new codebreakers: essays dedicated to David Kahn on the occasion of his 85th birthday*, Lecture Notes in Computer Science, 9100, Springer, 2015. ISBN 978-3-662-49300-7. See [30].

[106] Jim Salter, *A Chrome feature is creating enormous load on global root DNS servers* (2020). URL: https://arstechnica.com/gadgets/2020/08/a-chrome-feature-is-creating-enormous-load-on-global-root-dns-servers/. Citations in this document: §2.1.

[107] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, Niels Ferguson, *Performance comparison of the AES submissions*, in Second AES Candidate Conference (1999), 15–34. URL: https://www.schneier.com/academic/paperfiles/paper-aes-performance.pdf. Citations in this document: §2.6, §2.6, §2.7, §2.7.

[108] Michael Scott, *Re: NIST announces set of Elliptic Curves* (1999). URL: https://groups.google.com/forum/message/raw?msg=sci.crypt/mFMukSsORmI/FpbHDQ6hM_MJ. Citations in this document: §3.7.

[109] Senate Select Committee on Intelligence, *Unclassified summary: Involvement of NSA in the development of the Data Encryption Standard* (1978). URL: https://www.intelligence.senate.gov/sites/default/files/publications/95nsa.pdf. Citations in this document: §3.6.

[110] Dimitrios Sikeridis, Panos Kampanakis, Michael Devetsikiotis, *Post-quantum authentication in TLS 1.3: a performance study*, in NDSS 2020 [**3**] (2020). URL: https://eprint.iacr.org/2020/071. Citations in this document: §2.2, §2.4, §2.4, §2.4, §2.4, §2.4, §2.4, §2.4, §4, §2.4.

[111] Marc Stevens, *CWI cryptanalyst discovers new cryptographic attack variant in Flame spy malware* (2012). URL: https://www.cwi.nl/news/2012/cwi-cryptanalist-discovers-new-cryptographic-attack-variant-in-flame-spy-malware. Citations in this document: §3.2.

[112] Marc Stevens, Alexander Sotirov, Jacob Appelbaum, Arjen Lenstra, David Molnar, Dag Arne Osvik, Benne de Weger, *Short chosen-prefix collisions for MD5 and the creation of a rogue CA certificate*, in Crypto 2009 [**63**] (2009), 55–69. URL: https://iacr.org/archive/crypto2009/56770054/56770054.pdf. Citations in this document: §3.2, §3.2, §3.2.

[113] Nassim Nicholas Taleb, *Skin in the game: Hidden asymmetries in daily life*, Random House, 2018. ISBN 978-0425284629. Citations in this document: §3.8.

[114] Haixu Tang, Xiaoqian Jiang, Xiaofeng Wang, Shuang Wang, Heidi Sofia, Dov Fox, Kristin Lauter, Bradley Malin, Amalio Telenti, Li Xiong, Lucila Ohno-Machado, *Protecting genomic data analytics in the cloud: state of the art and opportunities*, BMC Medical Genomics **9** (2016), article 63. URL: https://bmcmedgenomics.biomedcentral.com/articles/10.1186/s12920-016-0224-3. Citations in this document: §2.5.

[115] Eran Tromer, Dag Arne Osvik, Adi Shamir, *Efficient cache attacks on AES, and countermeasures*, Journal of Cryptology **23** (2010), 37–71. URL: https://link.springer.com/article/10.1007/s00145-009-9049-y. Citations in this document: §1.

[116] Meltem Sönmez Turan, Kerry McKay, Çağdaş Çalık, Donghoon Chang, Lawrence Bassham, *Report on the first round of the NIST lightweight cryptography standardization process*, NISTIR 8268 (2019). URL: https://csrc.nist.gov/publications/detail/nistir/8268/final. Citations in this document: §1.1.

[117] Meltem Sönmez Turan, Ray Perlner, Lawrence Bassham, William Burr, Donghoon Chang, Shu-jen Chang, Morris Dworkin, John Kelsey, Souradyuti Paul, Rene Peralta, *Status report on the second round of the SHA-3 cryptographic hash algorithm competition*, NISTIR 7764 (2011). URL: https://csrc.nist.gov/publications/detail/nistir/7764/final. Citations in this document: §1.1.

[118] Visiting Committee on Advanced Technology of the National Institute of Standards and Technology, *NIST cryptographic standards and guidelines development process* (2014). URL: https://www.nist.gov/system/files/documents/2017/05/09/VCAT-Report-on-NIST-Cryptographic-Standards-and-Guidelines-Process.pdf. Citations in this document: §1, §1.1.

[119] Xiaoyun Wang, Hongbo Yu, *How to break MD5 and other hash functions*, in Eurocrypt 2005 [**44**] (2005), 19–35. URL: https://www.iacr.org/cryptodb/archive/2005/EUROCRYPT/2868/2868.pdf. Citations in this document: §3.2.

[120] Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, Shai Halevi (editors), *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, Vienna, Austria, October 24–28, 2016*, ACM, 2016. ISBN 978-1-4503-4139-4. See [31].

[121] Christian Wenzel-Benner, Jens Gräf, *XBX: eXternal Benchmarking eXtension for the SUPERCOP crypto benchmarking framework*, in CHES 2010 [**82**] (2010). URL: https://link.springer.com/chapter/10.1007/978-3-642-15031-9_20. Citations in this document: §2.10.

[122] Christian Wenzel-Benner, Jens Gräf, John Pham, Jens-Peter Kaps, *XBX benchmarking results January 2012* (2012). URL: https://csrc.nist.rip/groups/ST/hash/sha-3/Round3/March2012/documents/papers/WENZEL_BENNER_paper.pdf. Citations in this document: §2.10.

[123] Michael Wertheimer, *Encryption and the NSA role in international standards*, Notices of the AMS (2015), 165–167. URL: https://www.ams.org/notices/201502/rnoti-p165.pdf. Citations in this document: §3.7.

[124] Bas Westerbaan, *Sizing up post-quantum signatures* (2021). URL: https://web.archive.org/web/20211123165010/https://blog.cloudflare.com/sizing-up-post-quantum-signatures/. Citations in this document: §2.4, §2.4, §4.

[125] Michael J. Wiener, *Efficient DES key search*, Carleton University School of Computer Science Technical Report TR-244 (1994). URL: https://www.sites.google.com/site/michaeljameswiener/. Citations in this document: §1.

[126] John Young, *NSA FOIA documents on Joseph Meyer IEEE letter* (2010). URL: https://cryptome.org/0001/nsa-meyer.htm. Citations in this document: §3.6.

[127] Kim Zetter, *How a crypto 'backdoor' pitted the tech world against the NSA* (2013). URL: https://www.wired.com/2013/09/nsa-backdoor/. Citations in this document: §3.7.

A History of this paper

In May 2019, a few months after CAESAR finished, the Journal of Cryptology wrote to me describing plans for a special issue on CAESAR. The journal invited me to write an "overview article" regarding the CAESAR "process", including its "genesis", how the competition was "structured", etc.

The CAESAR procedures were based on various procedures that had been used in previous cryptographic competitions, but made various changes designed to reduce security risks. But how much could I reasonably expect the reader to know about competition procedures, and about the way that those procedures influence risks?

Ask an academic cryptographer how cryptographic risks are analyzed. You'll hear about what academic cryptographers are typically trained to do: search for proofs, and search for attacks, and search for cryptosystems that allow proofs and don't allow attacks. Ask what went wrong in the latest security disaster and you'll be told that the cryptosystems were proven in weak models, or that the attack search was too limited, so we need more funding for cryptographers to keep searching for proofs and attacks and cryptosystems.

All of this sounds reasonable. But cryptography also includes *choosing which systems to use*, for example via a competition. Doing a bad job here can create security disasters, perhaps even more easily than doing a bad job of searching for proofs or attacks or cryptosystems. How do we analyze the risks involved in various processes for selecting cryptographic algorithms?

There is very little literature directly on point. As the introduction says: “The *procedures* used in cryptographic competitions are frequently mentioned as introductory material, but not as a core topic of cryptographic risk analysis.” So I had to fill large gaps in the cryptographic literature, first to survey competition procedures and second to analyze their security impact. I had already done most of this in preparing for CAESAR, but had never written it up.

I considered structuring the paper chronologically, with the pre-CAESAR background followed by CAESAR followed by more recent competitions. But it quickly became clear that it was better to transpose the structure, using various competitions at the bottom level as case studies within higher-level analyses of long-lasting problems such as the allocation of cryptanalytic time.

It also quickly became clear *why* there was this gap in the literature. First, some of the techniques used to scientifically analyze security risks in algorithm-selection processes come from operations research, risk analysis, etc.—but these areas are not part of common cryptographic curricula. Second, the people writing papers in those areas have no reason to bother looking at cryptography: they have a wealth of high-impact problems to analyze that are more approachable than cryptography is. Third, cryptographers have a perverse incentive to actively *ensure* a steady drumbeat of disasters—see Section 3.8—whereas systematically addressing the most glaring cryptographic risks, including algorithm-selection risks, could cause the drumbeat to stop.

I finished the paper in September 2020 and sent it to the journal. I also sent it to colleagues and incorporated various feedback. I posted it in December 2020 and then incorporated further feedback. I was also studying further examples, and added extra material to Section 2.

In April 2021, seven months after submission, I received two reviews from the Journal of Cryptology, along with a request to revise the paper. I sent a revision to the journal in May 2021, along with comments on the reviews. The rest of this appendix is a copy of those comments, modulo automatic changes in reference numbers.

> `Reviewer #1: The paper is quite unusual in its form for JoC.`

It is unusual for a review to begin by commenting on form rather than content. The review seems to be trying to suggest that there is a problem with the form. However, it is completely unclear what this problem is supposed to be.

Any well-versed scientist knows that differences in the topic being studied often naturally lead to differences in the form of the study. A paper studying the effect of a vaccine will have a section explaining its experimental procedures; a paper modularizing FO QROM security proofs will not. I would presume that whatever is bothering the reviewer about the *form* of this paper is actually a consequence of the *topic* of this paper, although obviously I can't verify this given that the review doesn't elaborate.

> The paper discusses about cryptographic competitions
> (as its title says).

The review's audience—the journal editors—already know what the title is, so this review sentence is content-free. "This paper analyzes security risks in various competition procedures" would have communicated more information in fewer words.

> It is a good idea to have such paper.

If "such" merely means any paper that discusses cryptographic competitions, then such papers already exist: see, e.g., the references that discuss NISTPQC.

> Unfortunately, there is no clear goal in this paper.

Not true. The paper has a clear twofold goal, stated in the abstract: (1) to survey procedures that have been used in cryptographic competitions, and (2) to analyze the extent to which those procedures reduce security risks.

When I see a strange claim in a review, I try to figure out how the reviewer could have arrived at such a claim—in particular, whether I could improve the situation by clarifying something. But the abstract clearly and concisely states what the paper is doing:

> Competitions are widely viewed as the safest way to select cryptographic algorithms. This paper surveys procedures that have been used in cryptographic competitions, and analyzes the extent to which those procedures reduce security risks.

I have no idea how the reviewer could claim that there's "no clear goal in this paper".

> The paper is more a list of anecdotes

No. My best guess is that what the reviewer is trying to do here, modulo understandable confusion regarding English usage, is dismiss the examples given in the paper as "anecdotal evidence"—but that's still wrong.

As the Merriam–Webster dictionary puts it, anecdotal evidence is "evidence in the form of stories that people tell about what has happened to them". This paper instead emphasizes verifiable facts, backed by extensive citations. The paper also contains new *analysis* of the documented facts.

Perhaps the reviewer isn't actually trying to raise the *procedural* objection captured by the phrase "anecdotal evidence". Perhaps the reviewer is instead trying to raise a *content* objection, claiming that the examples—despite being verifiable—are non-representative. But this claim isn't justified in the review.

Paul Halmos, winner of the 1983 Steele Prize for Mathematical Exposition, continually gave examples in his talks and writings. He also wrote extensively regarding the benefits of doing this. See, e.g., page 324 of Halmos's "I want to be a mathematician", where Halmos described a research talk that he had given: "I was lucky: it turned out that there was a small concrete special case that contained within itself all the concepts, all the difficulties, and all the steps needed to understand and to overcome them. I focused my talk on that special case . . . [details of the example] . . . and I felt proud: I thought I succeeded in communicating a pretty problem and its satisfactory solution without getting bogged down in irrelevant analytic technicalities."

The conventional mathematical culture instead describes examples as being "weaker" than generalizations. In this culture, focusing on an example is a sign that one has failed to analyze the situation in more generality. Halmos continued by explaining the condescending reaction that his talk had received from another famous mathematician, clearly a mathematician coming from this culture.

Most areas of scientific research don't have the mathematician's confidence. Analyzing examples is then more than just an expository choice: it's well known to be essential for avoiding errors. Someone writing a new analysis of risks in procedures followed by humans can *try* to reason logically from a model of the procedures but should *of course* be checking one example after another in case the model is oversimplified. Perhaps what we're seeing from this reviewer is simply culture shock: "What's going on with all these examples of human activity? Where are the theorems?"

Or perhaps the issue is actually something different. A more detailed review might allow the reader to figure out what the reviewer's problem actually is.

I should note that Section 3.1 of the paper does have a data point that I would describe as anecdotal evidence, a paragraph explicitly reviewing my thought process regarding the number of rounds in Salsa20. This is part of the analysis leading up to "Submitters see what happened in previous competitions. This feedback loop keeps most security margins in a narrow range." The fact that most security margins are in a narrow range is a publicly verifiable fact, and I did not overstate my evidence for attributing this fact to the feedback loop from previous competitions. If there's going to be more investigation of why cryptographic security margins are so small and what can be done to make them larger, then surely a useful step would be to collect more of these data points. Medicine publishes what it calls "case studies" for much the same reason.

For comparison, the full version of the famous LPR paper claims that "All of the algebraic and algorithmic tools (including quantum computation) that we employ in our hardness reductions can also be brought to bear against SVP and other problems on ideal lattices. Yet despite considerable effort, no significant progress in attacking these problems has been made." There is no citation here: this is providing *anecdotal evidence* regarding the unpublished but supposedly "considerable" effort by the authors to attack their own new assumptions. Did reviewers force LPR to remove this unverifiable claim regarding the level of study of the central security question raised by their paper? Evidently not. Did reviewers mischaracterize the entire LPR paper as “more a list of anecdotes”? I doubt it.

> `(all of them being interesting)`

Hardy’s questions about a paper, according to Halmos: “is it true?, is it new?, is it interesting?”

> `without much structure.`

In fact, the paper has the following structure, and the submitted version had the same structure except that it had only two of the “machinery” subsections:

- Introduction
- Speed
  - The machinery of cryptographic performance advertising, part 1: measurements: part 2: confirmation bias; part 3: systematic exaggeration
  - Competitions for cryptographic performance
  - How AES performance was compared
  - The process of comparing cryptographic speeds
  - How AES performance was compared, part 2
  - Better benchmarking mechanisms
- Security
  - The complex relationship between speed and security
  - The complex relationship between speed and security, part 2
  - The overworked cryptanalyst
  - Cryptographic risk management beyond timing
  - The goal of limiting security
  - The difficulty of recognizing attackers
  - The goal of producing publications

This structure was, and is, displayed explicitly in the paper as section titles and subsection titles.

> `If I had to summerize the paper, I would say:`

If I wanted to summarize the paper, I would say “Competitions are widely viewed as the safest way to select cryptographic algorithms. This paper surveys procedures that have been used in cryptographic competitions, and analyzes the extent to which those procedures reduce security risks.” This is exactly what the abstract says.

If I wanted to summarize the paper in more detail than this, I would write “Cryptographic competitions are not a panacea”. I would summarize how DES failed, and how AES failed, and how SHA-3 failed, and so on, exactly as in the introduction.

Security risks in cryptographic competitions are a complicated, multi-faceted topic. At the time of submission this was already a 37-page survey. There’s tons of analysis that has never appeared before—hardly a surprise given the novelty of the topic. I chose a few points to highlight in the introduction, the same way that a book author chooses a few points to highlight in the introduction, but obviously such a complex topic will always have aspects beyond what can be summarized in an introduction.

Shortly after I posted the paper, summaries appeared online of the points that various readers found most interesting. These summaries were generally longer than the summary in this review, and, unlike this review, gave me the feeling that the readers had spent time reading the parts of the paper that they were interested in.

```
- cryptographic algorithms have been compared with
inconsistent performance metrics
```

The paper's subsections "How AES performance was compared", "The process of comparing cryptographic speeds", etc. cover many different things that go wrong with these comparisons: e.g., "The costs of cryptographic optimization—in human time or computer time—can be huge obstacles for submitters without serious optimization experience." It's baffling to see a review summarizing this as "inconsistent performance metrics".

```
(which the author solved with several projects
e.g. SUPERCOP);
```

Performance comparison spans 9 pages, 1/4 of the paper. Benchmarking is an important part of this, but certainly not the only part; see generally Figure 2.7. Structurally, the paper explains performance-comparison problems beyond benchmarking problems. For example, improved benchmarking doesn't solve the important problem quoted above: "The costs of cryptographic optimization—in human time or computer time—can be huge obstacles for submitters without serious optimization experience."

Within benchmarking, there have been several steps forward, and I think that the projects I've coauthored have made enough contributions to justify the half page spent describing them. This half page is immediately followed by a list of four examples of benchmarking issues not solved by SUPERCOP, with citations to 8 papers from other people working on these issues. So the paper certainly doesn't say that SUPERCOP solves all the benchmarking problems—it says exactly the opposite of this.

Given the journal's invitation to write a paper describing how CAESAR was run, I could easily have justified focusing on material explaining how CAESAR solved problems in how earlier competitions were run, including solutions to problems in previous procedures for benchmarking. However, the reality is that there are benchmarking problems that still aren't solved today, and broader performance-comparison problems that still aren't solved today. It's important for future competitions to keep working on this.

So this line of the review overstates, in two directions, the extent to which the paper says that the problems surveyed in this section are solved by existing work;

i.e., it understates how much work remains to be done. I've re-read everything, and I can't figure out how the reviewer arrived at this remark.

> - cryptographic algorithms have inconsistent security margins;

Well-known examples of different security margins show up as background for new analyses in the paper—but the paper's statement "this feedback loop keeps most security margins in a narrow range" is very much the opposite of the review's summary!

> - adveraries (NSA) may be invisible;

Yes, the paper says "NSA can secretly hire consultants to participate in the competitions and to try to weaken security in the same way that NSA would have. These consultants can deny NSA involvement, the same way Tuchman did."

However, the very next paragraph says that "it is instructive to look back at the extent to which the cryptographic community has failed to recognize NSA as an attacker, never mind the harder problem of recognizing others working with NSA as attackers". So the review understates the problem—it's not just NSA being invisible.

Note that the subsection title, "The difficulty of recognizing attackers", takes as few words as the review's summary but does a better job of capturing the scope of the problem.

> - (academic) designers have little incentive to make cryptographic algorithms perfect (boring).

It's true that Section 3.8 briefly mentions the lack of incentive *for* boring cryptography, but what it emphasizes is an incentive that actively works *against* boring cryptography. For example: "We all have a perverse incentive to stay in this situation, collectively creating a neverending series of cryptosystems failing in supposedly new and exciting ways, so that we can continue writing papers designing and analyzing the next systems."

Saying that there's an incentive *against* boring cryptography, and also little incentive for it, is much stronger than merely saying that there's little incentive for it. So the review is again understating what the paper says.

> (Maybe there are more I am missing.)

The list of four short items quoted above was supposed to be the reviewer's summary of the paper. Why would the reviewer be unsure regarding the level of completeness of the summary?

Of course the first theory to explore is that this is the paper's fault. But I've already quoted various clear statements from the paper that obviously aren't captured by the four items above.

> It would be useful to make those conclusions more visible.

In writing the paper I already spent considerable time thinking through a selection of points to highlight in the introduction of the paper, and a selection of subsection titles. My assessment is that supplementing and/or replacing those with any of the points listed by the reviewer would damage the paper.

> There are some aspects of the style of the paper which bug me.

See below.

> First of all, I think no \cite should play any grammatical role in sentences. Sentences should stay grammatically correct if we remove them all.

There are two different approaches in the literature: parenthetical citations and noun citations. Let's look at, e.g., "Obfuscating circuits via composite-order graded encoding", the first paper in the April 2021 volume of the journal, and search through that paper for brackets:

- "The second property has several different formulations, most notably virtual black box (VBB) and Indistinguishability Obfuscation (iO) [3]." The "[3]" is is grammatical if it's treated as parenthetical, but not if it's treated as a noun.
- "The first candidate general-purpose obfuscator has been introduced by Garg et al. [11]." The "[11]" is grammatical if it's treated as parenthetical. It's also grammatical if it's treated as a noun; this is an example of what's called "apposition" in English, and not having commas means that the appositive noun "[11]" restricts the meaning of "Garg et al.".
- "Their work and follow-ups such as [4, 7] relied on Graded Encoding Schemes (GES) [8, 10] which generalize the more traditional notion of multilinear maps." The "[4, 7]" is not grammatical if it's treated as parenthetical ("such as relied on"). It's grammatical if it's treated as a noun.

To summarize, taking a minute to check the journal immediately shows that the journal style doesn't impose the extreme rule of banning noun citations. Why, then, does a review adopt this rule? (The journal also doesn't impose the rule at the opposite extreme, demanding that sentences stay grammatically correct if each bracketed number is replaced by a noun.)

Beyond observing that the review's style rules don't match the journal style, let's take a moment to look at the merits of these two approaches.

What exactly does the "[3]" mean in "The second property has several different formulations, most notably virtual black box (VBB) and Indistinguishability Obfuscation (iO) [3]"? Is "[3]" being cited as introducing iO? Or as introducing VBB and iO? Or as asserting the notability of VBB and iO? Or as making the entire statement—the fact that the second property has several different formulations, notably VBB and iO?

For people allergic to obfuscation, let's also look at an example from the second paper in the same volume, "Bloom filter encryption and applications to efficient forward-secret 0-RTT key exchange", a paper that also uses noun citations and parenthetical citations: "One central ingredient to secure today's Internet is key exchange (KE) protocols with the most prominent and widely deployed instantiations thereof in the transport layer security (TLS) protocol [45]." Is "[45]" being cited as introducing TLS? Or as stating that TLS has the most prominently and widely deployed instantiations of KE? Or as making the entire statement?

The authors are not answering these reader questions. Some authors might claim that a parenthetical citation always means the whole preceding sentence—but checking "[45]" shows that this isn't what the second paper meant here. Other authors might claim that a parenthetical citation always means the narrowest possible scope—but checking "[3]" shows that this isn't what the first paper meant here. Maybe someone can come up with a rule that matches all the journal papers, but the reality is that readers don't know this rule.

The obvious way to answer the questions is to say more about what the cited work is doing:

> See [45] for the latest version of the TLS specification and for references to earlier versions.

Compressed version of an example from my paper:

> NIST refused to list speeds of the fast Serpent implementation from [84].

Trying to say the same things with parenthetical citations is more difficult. One has to come up with another way to describe the work being cited, such as using the author name to mean the author's paper:

> NIST refused to list speeds of the fast Serpent implementation from Osvik [84].

Even worse, the parenthetical "[84]" creates an ambiguity that didn't exist before: is "[84]" is a citation for Osvik's implementation, or is it a citation for NIST's refusal? For clarity one has to put the parenthetical citation in the middle of a clause:

> NIST refused to list speeds of the fast Serpent implementation that Osvik [84] had introduced.

As a separate issue, I think credit should be the primary driver of whether and how often the name of a cited author is mentioned in the text; I don't think a name should be given extra air time simply because some reviewer thinks that noun citations are bad.

Authors switching away from parenthetical citations are much less likely to create ambiguities than authors switching away from noun citations. The only harm of replacing "[3]" with "(see [3])" is the space consumption. "See" occurs a few dozen times in my paper, including a burst in the introduction. There's also a positive side of the extra typing: each time I type "see" I'm asking myself whether I should be saying more about the reference, and if so what exactly I should say. For example, sometimes I decided to write "[. . .] shows", and sometimes I decided to write "[. . .] claims".

To summarize, the review's top "style" complaint is trying to enforce a rule that (1) is not followed by the journal and (2) encourages bad writing—most commonly a lack of clarity. I've never seen an argument that switching from noun citations to parenthetical citations improves clarity or has other benefits for the reader. The widespread use of parenthetical citations seems easy to explain as minimizing work for an author adding a citation to a paper—just stick in a `\cite{...}` somewhere, no other text changes required—but this isn't a good idea if this means that many readers waste extra time checking references.

```
Second, scientific papers should avoid writing in a
letter/speech form, at the first person. It is a bit
annoying to read "I" all the time. Sentences like "I
want to believe that this is important"
```

Let's look at this quote in context:

> Sometimes speed competitions have a clear utilitarian purpose. [example] . . . Sometimes people participate in or watch speed competitions simply for the thrill. [example] . . . If I set a speed record for some computation, am I doing it just for the thrill? Does the speed record actually matter for users? Making software run faster is a large part of my research; I want to *believe* that this is important, and I have an incentive to exaggerate its importance. People who select my software for its performance have a similar incentive to exaggerate the importance of this performance as justification for their decisions. Perhaps there is clear evidence that the performance is important, or perhaps not.

This introductory material is followed immediately by the beginning of "The machinery of cryptographic performance advertising".

The incentive structures here are an important part of the risk analysis. What would it look like to communicate the same incentive structures without the word "I"? One answer would be to generalize:

> If researcher X sets a speed record for some computation, is X doing it just for the thrill?

But then I would expect complaints from researchers who think I'm attacking them for their misbehavior. Highlighting *my own* incentives straightforwardly avoids this problem, while clearly and concisely communicating the main point. When competent communication violates oversimplified rules regarding writing style, the problem is with the rules, not with the communication.

This paper began with the journal inviting me to describe the "genesis" etc. of CAESAR, precisely because of my position initiating and running CAESAR. Some of the facts relevant to this are most naturally expressed as first-person observations. Regarding “all the time”, the single paragraph quoted above already covers 10% of the “I” occurrences in a 37-page paper.

```
or "But wait a minute" are not appropriate.
```

At Crypto 2018, Rogaway and Zhang wrote “But wait: just what definition is it that we call *customary*?” Was this also inappropriate? What exactly was inappropriate about it? Or is the rule that “But wait” is appropriate while “But wait a minute” is not appropriate?

```
The abuse of repeated questions such as "Does the
speed record actually matter for users?" is not the
right style, in my opinion.
```

Abuse is improper by definition, but what exactly is the review claiming to be an “abuse”? When Rogaway and Zhang at Crypto 2018 asked “But wait: just what definition is it that we call *customary*?”, was this also an “abuse”?

(It took me under one minute on Google to find an example of a recent paper at a flagship IACR conference simultaneously violating the most obvious interpretations of two consecutive rules from this review.)

“Repeated” is puzzling. The question “Does the speed record actually matter for users?” appears exactly once in the paper. Perhaps the reviewer meant to refer to the fact that there are dozens of questions in the paper. Is the reviewer claiming that a few questions would be okay, but that dozens is too many for a 37-page paper?

It should be obvious that the number of questions raised by a paper depends on the content of the paper. It is not surprising to see many questions in a long paper on a new topic. As a matter of personal style, some authors prefer to avoid all question marks in their papers, but I often find question marks natural when there is a question to ask.

```
Probably the main problem with this submission is the
lack of structure.
```

“Without much structure” earlier in the review suggests that there’s *some* structure but not enough. “Lack of structure” here is more extreme, suggesting that there’s no structure at all: a shapeless mass of text floating in the void. In fact, the paper has an explicit structure, quoted above.

```
We have a list of (interesting) stories
```

In English, the word “stories” can be used positively. Random example, from the MEE-CBC paper at FSE 2016: “As a first contribution and motivation for our work, we provide new evidence of this latent problem by recounting the story of Amazon’s recently released s2n library, to which we add a new chapter.”

In this case, however, the review has already claimed that the paper is “more a list of anecdotes (all of them being interesting)”. The reader understands “a list of (interesting) stories" to be referring to these "anecdotes" again. So this sounds negative in context—but the most obvious interpretation of "anecdotes" is simply wrong in what it's claiming regarding the paper, and as explained above I'm unable to figure out what the reviewer's problem actually is.

It's also puzzling that the review keeps pointing to the paper's examples without ever acknowledging that the paper also has new analysis. This is yet another way that the review understates the contents of the paper.

> `which are put together around the theme of speed or security.`

In fact, the structure is much more detailed than simply a "theme of speed or security". It's puzzling to read a review that fails again and again to acknowledge the explicit structure in the paper.

> `But there is no conclusion`

Not true. For example, the statement "Cryptographic competitions are not a panacea" is a conclusion. This conclusion is presented immediately before its supporting data; this is one of several standard writing techniques.

> `and the objective of the paper is unclear.`

The objective of the paper is clear: "Competitions are widely viewed as the safest way to select cryptographic algorithms. This paper surveys procedures that have been used in cryptographic competitions, and analyzes the extent to which those procedures reduce security risks."

> `Therefore, I cannot recommend acceptance.`

This review contains several outright errors, and systematically understates the paper's contributions.

> `The part about speed is a bit lengthy.`

"There is overwhelming evidence of performance requirements—whether real or imagined—playing an important, perhaps dominant, role in cryptographic competitions." Spending half the paper on speed is perfectly appropriate.

> `Section 2.1 and 2.3 are a bit out of the scope.`

No, they aren't. (Note that this 2.1 is split into 2.1 and 2.2 in the current paper, and this 2.3 is now 2.4.)

Let's start again with what the abstract says the paper does: "Competitions are widely viewed as the safest way to select cryptographic algorithms. This paper surveys procedures that have been used in cryptographic competitions, and analyzes the extent to which those procedures reduce security risks."

The paper presents evidence regarding the weight of performance in these procedures and in security risks. For example, the recent attacks of [42] would not have occurred if Serpent had been selected as AES; NIST documents indicate that Serpent ("high security margin") would have been selected over Rijndael ("adequate security margin") if performance had been given lower weight in the AES competition procedures. The paper also presents evidence that this weight has been, and continues to be, amplified by incentives that produce a persistent pattern of inaccurate claims regarding performance requirements. All of this is fully within the stated scope.

The fact that some types of security risks are outside standard cryptographic education—how many cryptographers have experience analyzing error patterns? selection bias? incentive structures?—is not a valid scope argument. On the contrary, it is a reason for the paper to spend space ensuring that the material is comprehensible to an audience that has not seen this type of analysis before.

```
Some section numbering are not consecutive
(there is no 2.2 between 2.1 and 2.3, no 2.7 between
2.6 and 2.8).
```

Krantz's writing guide says "Some authors number their theorems from 1 to n, their definitions from 1 to k, their lemmas from 1 to p, their corollaries from 1 to r—each item having its own numbering system. . . . As a reader, I find this method maddening; the upshot is that I can never find anything." Steenrod's writing guide had already pointed out how difficult it is to "locate a desired reference number" if "the author numbers lemmas separately from theorems". Exactly the same comments apply to figures, subsections, etc.

Unifying the numbering system into a single linear order takes a few lines in LaTeX. In the submission, the reader reads 2.1 (a subsection), then 2.2 (a figure in that subsection), then 2.3 (a subsection), etc. In the current paper, the reader reads 2.1 (a subsection), then 2.2 (a subsection), then 2.3 (a figure in that subsection), then 2.4 (a subsection), etc. The consecutive numbering makes it easier for the reader to follow references to specific numbers.

```
The sentence in the second \item on p.8 is
grammatically broken.
```

There are three sentences there, and I don't see a problem with any of them. The middle sentence is the longest and, in brief, looks like this: "X here stated A and B, and X there claimed C, but X did D." Best guess is that the reviewer doesn't understand that English allows the commas. Writing "X here stated A and B and X there claimed C but X did D" would also be grammatical, but the parser state after "A and B and" would be expecting a continuation of the object by default, and would then have to spend time rewinding.

```
Reviewer #2: I think it is a must to include in the
special issue an article from competition manager.
```

This seems to be why the paper was invited in the first place.

```
However current text is blog-style
```

Review #2 here manages to be even more vague than review #1 claiming a lack of "structure". One can speculate endlessly regarding what the reviewer is thinking. Does the reviewer also think that a case study in medicine is "blog-style"?

```
and has some issues:
-Sect. 2.1-2.3  on cryptographic performance
advertisement is too long,
```

Too long for what? Is there a page limit? In the submitted 37-page paper, these subsections were slightly under 4 pages of text plus a full-page figure. In the current paper (39 pages before this appendix), the equivalent sections 2.1 through 2.4 cover an extra example in detail, so they're longer by slightly more than 1 page.

Is the claim here that, given the goal of the paper, some particular material within these subsections *shouldn't* be included? Or is the claim instead that some particular material can be communicated just as clearly with fewer words? Which material?

Above I quoted a paper on "efficient forward-secret 0-RTT key exchange" that just appeared in the journal. Allowing 1 extra round trip in the key exchange would make that paper uninteresting—it was already well known how to achieve forward secrecy (with less computation time and less bandwidth, under more solid security assumptions) with 1 round trip. The introduction of that paper claims that "0-RTT protocols are probably going to be used heavily". This is a pervasive pattern in the cryptographic literature: one paper after another claims, in the introduction, that cryptographic performance is important, as part of saying why the paper is interesting.

Evidently such claims are within the scope of cryptography. It is also within the scope of cryptography to study these claims more carefully, and to point out errors in such claims, and to analyze how these errors occurred. The importance of the claims for cryptography means that these analyses are also important.

Processes for cryptographic algorithm selection increase cryptographic risks by putting a high weight upon performance. This weight, in turn, is driven by continual claims regarding the importance of cryptographic performance. The community has no mechanisms in place to ensure the accuracy of these claims, and has incentives against setting up such mechanisms. The patterns here are central material in my paper and are backed by detailed examples.

If a cryptographer expects claims regarding the importance of cryptographic performance to be short, positive, and covered *en passant* as part of a paper's introduction, then it must be shocking to see a detailed study of the topic. But this doesn't make the topic less important to study; on the contrary!

```
probably can be summarized shorter
```

This is content-free. I can point to any section of any paper and, with zero work, claim that the section "probably can be summarized shorter".

> and also reducing Fig 2.2
> which takes 1 full page.

I appreciate the clear, constructive nature of these specific words in the review. However, when the reviewer claimed that this part was “too long”, I was expecting the followup to say something about the text, not to complain about the size of the figure!

Reducing the figure would damage readability. The text points not just to the large-scale patterns in the figure but also to small-scale differences in the figure. The small-scale differences are already tricky to see at full size and would be much more difficult to see if the figure is reduced. What precisely is the benefit supposed to be?

> -The bulk of the text is about speed comparison

Not true. In the submitted paper, speed is 13.8 pages (including the picture), while the text as a whole is 28.5 pages (not including 8 pages of references). 13.8 is not “the bulk” of 28.5. The speed section is only marginally longer than the security section.

> which is important but other aspects seem to be overlooked.

See below.

> For example, competitions often go for new areas

Already covered in the paper: “Comparing the symmetric competitions shows trends towards larger and more complex inputs and outputs in the cryptographic algorithm interfaces. DES has a 64-bit block size; AES has a 128-bit block size. Stream ciphers encrypt longer messages. Hash functions hash longer messages. Authenticated ciphers include authentication tags in ciphertexts, and optionally authenticate another input. Many NISTLWC submissions support hashing and authenticated encryption, sharing resources between these functions. This does not mean that complexity is a goal per se: symmetric algorithms with larger interfaces often reach levels of efficiency that seem hard to achieve with smaller interfaces, and there are some security arguments for larger interfaces. See generally [21, Section 2].”

> where there are no well established standards

This is too vague to evaluate. Depending on how one defines each “area” and what counts as “well established”, one can say that every competition has broken new ground; or, at the opposite extreme, one can say that every competition has been in an “area” where there were already “well established standards”. DES is not an exception here—one simply has to declare that, e.g., Hagelin machines were de-facto “standards”.

> (ex. post-quantum crypto).

NISTPQC is one of the competitions analyzed in the paper. "Post-quantum authentication in TLS 1.3: a performance study" is a running example in Sections 2.1 and 2.3 of the submission (and now in 2.2 and 2.4). Still no idea what the reviewer claims is overlooked in the paper.

> `In such cases a lot of new things are learnt during the competition.`

Already covered in the paper: "A traditional report on the CAESAR competition would say that it produced many papers, advancing researchers' understanding of security and performance, building a foundation for the next generation of papers on symmetric cryptology."

> `So one of the key aspects is to what extent this knowledge can be used to improve the competing algorithms or we have to wait 5-10 years till the next competition.`

Already covered in the paper: "At an NBS workshop in 1976, before DES was approved as a standard, Diffie (in joint work with Hellman; see [50, page 77, 'Cost of larger key']) proposed modifying the DES key schedule to use a longer key. Representatives of Collins Radio and Motorola objected to this proposal, saying that DES is 'close to the maximum that could be implemented on a chip with present technology' and that a manufacturing delay 'of one to two years might be encountered if a longer key were required'."

Another example from the paper: "In its final AES report [89, Section 2.5], NIST had complained that requests to consider a different number of rounds for an AES submission 'would impact the large amount of performance analysis' that had already been done, since 'performance data for the modified algorithm would need to be either estimated or performed again'. It wasn't realistic to expect all the authors of performance-comparison papers to integrate new functions into their private benchmarking procedures and update their papers accordingly."

Both of these are examples of requests to replace an algorithm with a different algorithm identified during the competition. The first request was rejected for (claimed) performance reasons, and the second request was rejected for meta-reasons regarding the benchmarking process.

Competitions have often allowed "tweaks" in submissions, and it's fascinating to see the extremes to which the "tweak" concept has been pushed in NISTPQC. Certainly it would fit the scope of the paper to evaluate the security risks in different "tweak" procedures—there's the hope that allowing improvements to submissions will improve the competition output, vs. the damage that instability does to security evaluation and sometimes to performance evaluation—but I think the data points that NISTPQC is generating will need years of followup analysis.

> `-One aspect that could be more reflected in the paper is the growing number of submissions which goes beyond community's ability to evaluate`

This is covered too: "**3.3. The overworked cryptanalyst.** . . . This problem is more severe for competitions having more submissions—and for competitions having more complicated submissions." There's a figure charting the number of submissions. There's a quote from the NESSIE project comparing the review time spent on the AES finalists to the review time spent on DES.

Claiming that something "could be more reflected" is content-free, and doesn't justify the claim that something was overlooked.

```
and in general low motivation/reward for the analysis of
freshly designed primitives.
```

"Low" is a matter of perspective. The general issue of cryptanalysts deciding which primitives to attack is certainly covered.

```
-Describe the processes of the CAESAR competition  more
```

What exactly does the reviewer think is missing?

```
(currently
only two pages 23-24)?
```

Not true. Beyond pages 23–24, page 9 of the submission describes CAESAR's "use cases" and portfolio; page 16 describes CAESAR's software requirements for benchmarking; page 21 describes CAESAR's timeline; and page 27 describes CAESAR's anti-sabotage strategy.

Many more portions of the paper are describing analyses that took place as part of the preparation for CAESAR. The analyses usually don't mention CAESAR because their content isn't specific to CAESAR.

```
Conclusions,
```

Already there. For example, as noted above, the statement "Cryptographic competitions are not a panacea" is a conclusion, presented immediately before its supporting data.

```
lessons learnt,
```

How would a "lesson" not also be a "conclusion"? Redundant request.

```
what could be done differently.
```

Already there. The paper is full of analyses of competition options, including some open questions. For example: "Would an attacker be able to sneak a weak algorithm through a committee full of experts? Would it be able to sneak a weak algorithm through a competition that simply takes the fastest unbroken algorithm? These are interesting questions to analyze."

```
Are the selected primitives used by the industry?
```

Salsa20 and its variant ChaCha20 are very widely used today. In 2010, two years after eSTREAM selected Salsa20, was this reviewer asking about Salsa20 usage? I'm aware of various deployment efforts for CAESAR portfolio members, but I think it's premature for the paper to cover this.

```
-The whole Sect.3.8 is dubious,
```

This sounds like the reviewer is disputing something. What, precisely?

```
these arguments can be applied to many areas of human
activity
```

Let's look at, e.g., the following statement: "Imagine a competition requiring every cipher to have twice as many rounds as it seems to need. This would make typical attack improvements less scary, and would eliminate most—although not all—cipher breaks. This would make papers harder to publish." How precisely is the reviewer claiming that this statement "can be applied to many areas of human activity"?

Even assuming, arguendo, that the statement is more broadly applicable, how does this make the section "dubious"?

```
(ex.
why do we have to buy new laptops every 4 years).
```

I understand this part of the review as pointing to an analogy between the laptop manufacturer's incentive to keep producing laptops and the academic cryptographer's incentive to keep producing papers. Let's look at some ways that this analogy fails.

In cryptography, as the paper says, it's "against community standards to blame the designers rather than blaming the broken system". This is not true for laptops. Laptop failures—e.g., battery failures—are systematically tracked *and the manufacturers are systematically blamed for the failures*, for example with Consumer Reports concluding that "Apple laptops experience the fewest breakdowns".

Failures are the centerpiece of cryptographic advertising. As the paper says, "papers and grant proposals on improved attacks and improved cryptosystems and security proofs habitually explain their importance by citing recent failures". Do laptop manufacturers habitually explain the importance of new laptops by citing recent failures? No. Sure, they'll sometimes point to reliability features, but there's much more advertising of connectivity, disk space, screens, CPUs, GPUs, etc. As Dell says: "keeps you connected", "safekeeping all your digital content", "vivid displays", "the performance you demand", "high-performance gaming", etc.

Eliminating cryptographic failures poses a clear existential threat. As the paper says: "Won't the users say at some point that they have exactly the right cryptographic operations and don't need further input from us? It's safer for us if our core cryptography keeps failing." For laptops, *maybe* users will decide they have all the connectivity and disk space and so on that they want, and *maybe* laptop manufacturers will have no way to maintain a laptop market except through engineering the laptops to fail, but it's hard to argue that this failure incentive is as powerful for laptops as it is for cryptographers.

That's already three examples of things stated in this section of the paper that are correct for cryptography but that, if stated for laptops, would range from highly questionable to clearly incorrect.

> `I would blame cryptographers the least for something like this,`

So we're supposed to believe that, across "many areas of human activity" including cryptography and laptop manufacturing, cryptographers are "the least" to blame for failures? That's a remarkably strong claim, and I don't see the evidence for it.

What's particularly puzzling is how the review jumps from

- claiming that this section is "dubious" to
- claiming that what the section says is also true for laptop manufacturers and then to
- claiming that there's an important difference here between cryptographers and laptop manufacturers.

The paper focuses on cryptography. I don't see how the review is disputing what the paper says. I don't see how the review is making a case for extending the statements to cover laptop manufacturing—on the contrary, this is clearly not as simple as the review suggests.

> `since they care about their reputation.`

This is a very weak statement. I don't see how this statement supports the "least" comparison earlier in the sentence, and I don't see how it's disputing anything in the paper.

> `-It would be good to finish the paper with the  conclusions section.`

See `https://cr.yp.to/writing/devil-conclusions.html` and the quotes therein, including Goldreich's comment "Certainly, the inclusion of a conclusion section should not be the default".

> `Associate Editor: Thank you for submitting your work to JoC CAESAR special issue. The review process has been completed, and the review comments can be found below. At this time, given their recommendation and comments, we made a decision of ``Revise''.`

I request that the journal stop using these two reviewers for this paper.

```
Please revise your paper reflecting review comments.
```

Compared to the September 2020 version of the paper: The biggest change is an extra example, adding slightly more than a page to Section 2.1, which is now split into two subsections. There are also smaller improvements at many spots in the paper.

```
In
particular, please address the comments pointing out the lack
of structure
```

Addressed above. The paper already had a detailed and explicitly displayed structure, contrary to the claims from the first reviewer.

```
and the length of Sect. 2.1-2.3.
```

Addressed above. These are important sections, now extended for an extra example as noted above.

```
Please address
other comments as well, and if you cannot reflect some of the
comments, please mention them in the reply letter.
```

Each comment is addressed above. Each suggestion that I did not take has a rationale stated above for not taking it.

```
Thank you once again for submitting your work to JoC CAESAR
special issue. We look forward to receiving the revision soon.
```

Here you go!