Towards post-quantum symmetric cryptography

John Gregory Underhill[1] and Stiepan Aurélien Kovac[2], and Xenia Bogomolec[3]

1) *itk AVtobvS SARL, ch. de Monséjour 2, 1700 Fribourg, Switzerland*
john.underhill@protonmail.com · http://itk.swiss
2) *itk AVtobvS SARL, ch. de Monséjour 2, 1700 Fribourg, Switzerland*
stie@itk.swiss · http://itk.swiss
3) *SOC Hessische Landesbank, 60311 Frankfurt am Main, Germany*
indigomind@protonmail.com · http://coder.tjingwan.com

Abstract. With this work, we intend on demonstrating the need for improvements to the currently standardized AES family of cryptosystems, and provide a solution that meets the requirements of long-term security in the rapidly evolving threat landscape. The solution proposed is flexible, dramatically increases the potential security of the cipher, and strongly mitigates many of the most serious attacks on the AES family of cryptosystems. Further, our solution can be easily integrated into existing AES cryptosystem deployments, with only a few small changes required, thus preserving the large investments in this cipher both in hardware and software.

Keywords: Symmetric · cryptography · quantum · AES

1 State of the art

1.1 AES

The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data. The AES selection competition was held in 1998 by the National Institute of Science and Technology (NIST), which after a period of study from the academic community, a subset of the Rijndael family of symmetric ciphers was selected as the AES competition winner.

In 2001 the adoption of AES as a replacement for the DES family of ciphers, was formally proposed with the release of the FIPS document FIPS PUB 197 [1]. Subsequently, the AES family of cryptosystems has been adopted by countries around the world, and has been near universally implemented, and is now the most widely used and critically important symmetric encryption scheme in use today.

AES is in fact, implemented on most of the computers in the world, and is the backbone of the worlds secure communications systems, not only as the primary encryption cipher used by e-commerce and secure communications applications, but also in the many and various diverse use case scenarios such as VPN technology, disk drive encryption, and secure databases.

It is for this reason, that this technology must be kept secure against future threats, for if a serious break in AES were ever discovered, the result would be nothing short of catastrophic, and result in the total and absolute compromise of the worlds secure communications infrastructure.

1.2 The need for change

In cryptography we adjust a ciphers strength according to what we consider to be its resistance to attack. For example, if we know that 64 bits of key can be broken, we set the key length to be at least 2n, or 3n that size. If we know that 10 rounds of a rounds-based iterative block cipher can be broken, we set that cipher to twenty or even thirty rounds. This is an assurance we provide that the technology is well beyond the current and expected future technological capabilities required to break that cipher.

Rijndael was not chosen for its security properties, but largely for reasons of its superior performance. Other leading candidates; Serpent and Twofish each scored significantly higher in their security evaluations, with it estimated that Rijndael variants would require being set at between 18 to 24 rounds only to equal the estimated security of the Twofish family of ciphers [2] (rather than the 10 to 14 rounds defined by NIST), and Twofish was in turn considered to be significantly less secure than the Serpent cipher.

The last prolonged and intensive public study made of Rijndael was during the AES standardization competition, and that was nearly 20 years ago. Since that time, many advances in cryptanalysis and computing technology have been made, the rate of technological progress has accelerated, and the immediate future promises to provide changes to our technological capabilities that far exceed what was even considered possible just a few years ago.

Since the AES selection process, many new forms of attack have been discovered and others greatly improved upon; some side-channel attacks that target subtle variations in a cipher's execution timing, have been proven capable of breaking AES under real world conditions [3][4][5]. Other attacks that target the differentially-weak key schedule [6][7], particularly in AES-256, have also raised serious questions about the suitability of AES in a long-term security context.

The authors of one of these related subkey attacks, which has to date broken 8 out of 10 rounds of AES-128 and 11 of 14 rounds of AES-256, pose the question of the currently understood AES safety margins [7]:

"While neither AES-128 nor AES-256 can be directly broken by these attacks, the fact that their hybrid (which combines the smaller number of rounds from AES-128 along with the larger key size from AES-256) can be broken with such a low complexity raises serious concern about the remaining safety margin offered by the AES family of cryptosystems."

Though these attacks are currently not known to be able to break a full 10, 12, or 14 round version of AES, they do prove two things conclusively: that the weak internal key expansion function is a serious and exploitable flaw in the cipher design, and that the number of mixing rounds is set too low.

Bruce Schneier has made public comments to this effect, repeatedly urging the community to consider increasing the number of rounds, and acknowledging this serious flaw in the ciphers design [8]:

"Cryptography is all about safety margins. If you can break n round of a cipher, you design it with 2n or 3n rounds. What we're learning is that the safety margin of AES is much less than previously believed. And while there is no reason to scrap AES in favor of another algorithm, NIST should increase the number of rounds of all three

AES variants. At this point, I suggest AES-128 at 16 rounds, AES-192 at 20 rounds, and AES-256 at 28 rounds. Or maybe even more; we don't want to be revising the standard again and again."

It must also be considered, that these are only the attacks that have been discovered in the academic community and as such are publicly known. For example, the NSA likely spends more on cryptanalysis than all of the universities in the world combined, they employ many of the best cryptanalysts, cryptologists, mathematicians and engineers in the world, and have been actively engaged in trying to break AES for nearly twenty years.

Quantum computers are now a reality. Engineering problems are being solved, working prototypes are being built, and their capabilities are rapidly improving. Many of the largest companies in the technology sector are now investing billions of dollars into these emerging technologies, along with state agencies from the most powerful countries in the world.

Some estimates are that these machines may be able to break the most widely used asymmetric ciphers in as little as five years [9].

They pose a threat to symmetric ciphers as well, with the potential to halve the key space, making the 128-bit version of AES effectively obsolete.

A new paper proposes a quantum algebraic attack against AES using Boolean equation solving [10] estimates that even greater reductions to the key space may be possible.

A lesson we have learned time and again, is that attacks only improve, and new attacks against symmetric cryptography using quantum computers will almost certainly be discovered once these computers and their properties are better understood.

Lastly, we must consider what we can not predict, the *unknown unknowns*. Some experts believe that we are within twenty years of the technological singularity; the point at which we create an independent machine consciousness more powerful than our own. Beyond this point, it will quickly become impossible to predict the future evolution of our technological capabilities, for by its very definition, we will have created an alien intelligence, which by the end of the century could evolve to become many orders of magnitude more powerful than human consciousness.

The definition of long-term security has changed in the last ten years. Historically, for data to be considered secure it only required that it should remain unreadable for so long as that information was relevant, typically ten to twenty years.

A great deal of evidence has been accumulating that strongly indicates that state sponsored intelligence agencies have become engaged in the collection and long-term storage of much of the worlds secure communications traffic. In parallel, laws that cater for the evolution of genetic medicine, ask for lifetime data protection, such as Germany's patient data protection law ("even after death of the patient"), setting a precedent in this domain to the European Union's GDPR, that now sees global effects.

It is in response to this change in the threat landscape, that we must in turn create and adopt more powerful and flexible encryption technology, to ensure that data is not protected only for the foreseeable future, but beyond what can be estimated or predicted given our current knowledge. We must now answer this present and

evolving threat, and attempt to ensure that sensitive information can be kept secure for an entire lifetime.

1.3 The rules of the game

Nearly twenty years after the standardization of AES; created as a replacement for the then popular DES family of ciphers, we can still find DES in widespread use. DES and the strengthened version of the cipher, Triple DES, are still widely used in the electronic payment industry, various software applications, aging TLS configurations and VPNs [11] and integrated into popular browser software.

This failure to update our systems and software to the currently held stronger standards of encryption technology is well understood, and supported by historical trends in the software industry.

The standardization process for a replacement for AES, if judging by past standardization efforts, (including the NIST Post Quantum asymmetric call, which could take three to five years to complete), could take as long as five years.

The actual changeover to a new symmetric cryptosystem given that AES is near universally implemented, and integrated into every level of our communications systems and encryption software componentry, could take another twenty years to be realized on such a massive scale.

Clearly this is completely unacceptable, particularly given the ample evidence that state agencies have begun harvesting the worlds secure communications traffic and can store that data indefinitely for future analysis.

There is more than sufficient cause to believe that AES is not as secure as we once believed it to be, and that given the massive investments into cryptanalysis by state funded agencies, and that breaking the AES cryptosystem represents the most fundamentally important target of their efforts, it becomes clear that improvements to the AES design that counter known weaknesses and provide greater security is becoming of paramount importance.

A simple formula demonstrates this:

$$x + y <= z$$

If **x**, the number of years we require a cipher to be secure, added to **y**, the time it takes to deploy a new cipher, exceeds **z**, the amount of time before a cryptanalytic or technological breakthrough might occur, or beyond the time when we may reasonably predict the future state of technology, then we have surpassed the secure lifetime of that cryptographic cipher. We believe that we have already passed that point in time.

The history of our industry is littered with broken ciphers and cryptographic protocols, in fact the history of science is one of theories, once thought concrete and inviolable, swept aside in the advent of new discovery. Newtons law of gravity held for nearly two hundred and fifty years, until a patent clerk in Switzerland proved it to be flawed.

We need to learn from that history, and we need to prepare for what promises to be an uncertain and increasingly unpredictable future, because the security of our communications systems now represents the security of nations, the sustainability of free and democratic states, and the security of all the people of the world.

2 A way forward

2.1 eAES

What we propose is an intermediate solution. If in fact we have exceeded the reasonable security lifetime of the existing form of the AES cryptosystems, and a new solution can not be widely implemented before such time as these ciphers might be broken, then we believe that an interim solution is the best current option.

This solution involves two steps:

1. The replacement of the differentially-weak internal key expansion function, the 'key schedule', with a cryptographically-strong pseudo random generator.
2. Increasing the number of transformation rounds to at least **2n**, or at least twice the best-known cryptanalytic attack against the AES ciphers.

The key schedule is essentially a key expansion function, that expands a small input cipher key, into a much larger internal array of round-keys, used by the ciphers transformation function to create a cipher-text output unique to that key. In the Rijndael cipher, this key expansion function is the weakest part of the construction, it does not produce cryptographic-quality output, and has been the target of several serious related-subkey and timing-based attacks on the cipher. We propose that this function be replaced with a cryptographically-strong pseudo-random generator.

We have produced a model implementation with this change; **RHX** (and the companion AES-NI based **AHX**) [12], which uses either the **HKDF(SHA2) Expand** key derivation function, or a choice of the Keccak based **cSHAKE XOF** function. We have also produced an example C based primitive **RSX** [13] which uses a fixed implementation of the Keccak extended output function **cSHAKE-256**. Both implementations use keyed pseudo-random generators that are widely accepted as producing highly diffused output and are considered to be strong cryptographic-quality generators.

Besides producing a more cryptographically secure output, (used to generate Rijndael's internal rounds sub-key array), replacing the key schedule with a cryptographically-strong generator also allows for the safe addition of more transformation rounds by securely generating the required longer subkey array. Using an existing and well-regarded strong generator also eliminates the need to modify the existing key expansion function, and replacing it with an *ad hoc* adjustment which could prove to be insecure, and would require substantial study before it could be used safely.

We have chosen both the HKDF(SHA2) Expand function and the Keccak based cSHAKE, because these are both widely regarded within the cryptographic community as being cryptographically-strong pseudo-random generators.

The HKDF version of these ciphers is inter-transitional, that until Keccak and its cSHAKE derivative have seen more widescale usage, and hence more scrutiny from our community, that some implementors may prefer the HKDF generator.

The HKDF variant can use either the SHA2-256 or the SHA2-512 version of the hash function as the primary pseudo-random function, and the cSHAKE variants have the cSHAKE-256 and the [experimental] cSHAKE-512 options in our C++ implementations [12].

Each variant will produce a completely different output cipher-text, and so must be considered as unique versions of the cipher extension. Additionally, we have added an information string (the distribution code property), that can be set in the cipher implementations as a cipher-tweak, this user definable string can be used to safely produce a unique cipher-text output.

By using these variations, and in fact a flexible model promoting interchangeability throughout our library implementation, we aspire to create a real-time upwardly-flexible security paradigm. One in which within the context of a broader domain-based communications system, the security of data transmissions can be actively modified during a hand-shake negotiation; parameters can be set at run-time, even the ciphers and protocols can be interchanged, guaranteeing the best possible security profile can be achieved and maintained.

The rate of technological progression has accelerated dramatically over the last century, and continues to expand, the advent of strong AI coupled with emerging quantum-based computing systems may serve to vastly accelerate our technological development, and with this in mind, we believe that a security-flexible core cryptographic library is essential to maintaining sustainable long-term security.

We also propose an increase in the number of transformation rounds; the number of mixing cycles applied to the state.

We will not consider the AES-128 or AES-192 members of the AES family of cryptosystems, because currently known attacks using quantum computers [10][14] will one day be able to break these variants, rendering them thus unsuitable for the purposes of long-term security.

The **eAES**, (the formal name of the proposed extension to AES) **256-bit** variant is set to **22 rounds**, or twice the known number of rounds broken by related subkey attacks. This is an increase of 8 rounds, from the 14 rounds used currently by the NIST standardized form of AES-256.

We have produced a **512-bit** key variant, that is set to **30 rounds**. There is a great deal of resistance within our industry to the use of 512-bit keys, many cryptographers consider them as an unnecessary key length, but one must consider what these assumptions are predicated upon; that known quantum attacks that halve the key space will never be improved upon, that no new attacks quantum or cryptanalytic based will ever be discovered, and that we can accurately predict the future of our technological development for at least fifty years.

We believe the fault in these arguments to be clear and self-evident, and that if we are to establish real long-term data protections, we must become better at anticipating the unknown.

The library also contains a 1024-bit variant which is set to 38 rounds, provided for the purposes of future experimentation.

What we are bringing to your consideration, is a working and provably-secure solution to the question of the AES ciphers sustainable long-term security. It would use the existing rounds function, the core component of this cipher, and so preserve the large investments that have been made in this technology, while strongly mitigating many of the most serious known attacks against the AES family of cryptosystems.

It would require only that existing implementations substituted the key schedule's expansion function with a cryptographically-secure alternative, and provide the

increase in rounds necessary to restore the ciphers current security margins to acceptable levels.
The performance penalty incurred by the key schedule when using a cryptographically-strong generator is a negligible one-time penalty for each encrypted stream and so does not significantly impact the transformation of medium to large data sets.

The increase in rounds will have a small effect on the speed of the cipher, but this must be considered as a reasonable cost, required by the necessary increase in the security of the cipher. We have been using AES for almost twenty years, and so transitioning to versions with increased security should not be wholly unexpected, and the relative cost incurred is still negligible, especially when considering the enormous increase of hardware speeds and technologies such as embedded instructions, that have more than compensated for this small loss in performance.

2.1 Pseudo code

The AES rounds function:

```
Transform(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin

 byte state[4,Nb]
 state = in
 AddRoundKey(state, w[0, Nb-1])

 for round = 1 step 1 to Nr–1
   SubBytes(state)
   ShiftRows(state)
   MixColumns(state)
   AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
 end for

 SubBytes(state)
 ShiftRows(state)
 AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
 out = state

end
```

Figure A.1: Note that the rounds function is identical to the AES specification. The number of rounds processed in the main loop is determined by the value of **Nr**.

The secure key expansion function:

```
SecureExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], word Nr)
begin

 word ks = Nb * (Nr + 1) * 4
 byte tmp[ks]
 tmp = HKDF-Expand(key, dc, ks)
```

```
 while (i < ks)
   w[i/4] = word(tmp[i], tmp[i+1], tmp[i+2], tmp[i+3])
   i = i+4
 end while

end
```

Figure A.2: The octet key size **ks** is calculated as the ciphers block size in 32-bit words **Nb**, times the number of rounds **Nr** + 1, times the number of octets in each 32-bit word.
The temporary array of bytes **tmp** is generated by HKDF Expand (or alternatively cSHAKE) using the input cipher key and the optional distribution code **dc** arrays as input. This array of octet sized integers is then converted to 32-bit words and added to the rounds subkey array **w**.

2.2 Horizon 2040 and beyond

We believe the future of symmetric ciphers is in authenticated stream ciphers. They are fast, trivial to parallelize, easy to use and thus less likely to incur implementation mistakes, and as we have learned, authentication has become an essential aspect to ensuring data integrity.

We have implemented several authenticated stream ciphers based on the Threefish and ChaCha ciphers, which can use either the Keccak based **KMAC**, or **HMAC(SHA2)** message authentication code generators, the option selected through a constructor setting. These ciphers have been strengthened in keeping with our mission to provide real and sustainable long-term security, and are implemented in C++ using CPU intrinsics and optional multi-threaded parallelization [12].

These stream cipher primitives have also been implemented with 512-bit keys, (and in the case of Threefish, a 1024-bit key), and an increase to the number of permutation rounds in the 512-bit and 1024-bit key versions, in accordance to our currently projected security levels that we feel will be required for periods exceeding the 2040 horizon.

2.3 Cryptanalysis

A first cryptanalysis of the suggested engineering changes to AES, done by researchers at Kudelski AG in Chéseaux, Switzerland, is provided at the following URL : https://git.fsfe.org/Stie/Pqsym/raw/branch/master/2019-04-29-Kudelski-AHX-Analysis-v01.05.pdf. The use of the term AHX throughout that analysis refers to the Intel-optimized implementation of eAES available under [12].

2.4 Future work

A performance analysis of the AHX implementation of the suggested changes to AES has been published on: https://git.fsfe.org/Stie/Pqsym/src/branch/master/Performance. Its authors, the UAS Western Switzerland in Fribourg, identified that it could run even faster on servers if using assembly language. Likewise and as has been suggested by Kudelski in their report, physical attacks on hardware implementations of eAES will be studied and generally, a deeper cryptanalysis will be done from that starting point.

2.5 Acknowledgements

We would like to thank the Swiss federal government for funding the work by the UAS Fribourg, likewise providing support in the CEuniX project that led to this work.

References

1) FIPS 197: ADVANCED ENCRYPTION STANDARD (AES)
NIST, November 2001
https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.197.pdf

2) Improved Cryptanalysis of Rijndael
Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Mike Stay, David Wagner, and Doug Whiting, 2000
https://www.schneier.com/academic/paperfiles/paper-rijndael.pdf

3) A Diagonal Fault Attack on the Advanced Encryption Standard
Dhiman Saha, Debdeep Mukhopadhyay, and Dipanwita RoyChowdhury, 2009
https://eprint.iacr.org/2009/581.pdf

4) Cache Games – Bringing Access-Based Cache Attacks on AES to Practice
Endre Bangerter, David Gullasch, Stephan Krenn, 2010
https://eprint.iacr.org/2010/594.pdf

5) Design, Implementation and Performance Analysis of Highly Efficient Algorithms for AES Key Retrieval in Cache Access Attacks
Ashokkumar C, Ravi Prakash Giri, Bernard Menezes, 2017
https://ieeexplore.ieee.org/document/7467359

6) Super-Sbox Cryptanalysis: Improved Attacks for AES-like permutations
Henri Gilbert and Thomas Peyrin, 2009
https://eprint.iacr.org/2009/531.pdf

7) Key Recovery Attacks of Practical Complexity on AES Variants With Up To 10 Rounds
Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir, 2009
https://eprint.iacr.org/2009/374.pdf

8) Another New AES Attack
Schneier on Security, July 30, 2009
https://www.schneier.com/blog/archives/2009/07/another_new_aes.html

9) IBM warns of instant breaking of encryption by quantum computers: 'Move your data today'
https://www.zdnet.com/article/ibm-warns-of-instant-breaking-of-encryption-by-quantum-computers-move-your-data-today/
Arvind Krishna, director of IBM Research, May 18, 2018

10) Quantum Algorithms for Boolean Equation Solving and Quantum Algebraic Attack on Cryptosystems
Yu-Ao Chen and Xiao-Shan Gao, 2018
https://arxiv.org/pdf/1712.06239.pdf

11) RFC 8221: ESP and AH Algorithm Requirements, p. 7
IETF, October 2017

https://tools.ietf.org/html/rfc8221#page-7

12) The CEX Cryptographic library in C++
John G. Underhill, last retrieved: November 23, 2018
https://github.com/Steppenwolfe65/CEX

13) RSX: Rijndael-cSHAKE hybrid
John G. Underhill, last retrieved: November 23, 2018
https://github.com/Steppenwolfe65/RSX

14) A fast quantum mechanical algorithm for database search
Lov K. Grover, 1996
https://arxiv.org/pdf/quant-ph/9605043.pdf