Energy-Efficient Cryptographic Engineering Paradigm

Marine Minier, Raphael Phan

Energy-Efficient Cryptographic Engineering Paradigm*

Marine Minier[1] and Raphael C.-W. Phan[2]

[1] Université de Lyon, INRIA
INSA-Lyon, CITI, F-69621, Villeurbanne, France
Marine.Minier@insa-lyon.fr
[2] Loughborough University
Electronic, Electrical & Systems Engineering
LE11 3TU, Leicestershire, UK
R.Phan@lboro.ac.uk

Abstract. We motivate the notion of green cryptographic engineering, wherein we discuss several approaches to energy minimization or energy efficient cryptographic processes. We propose the amortization of computations paradigm in the design of cryptographic schemes; this paradigm can be used in line with existing approaches. We describe an example structure that exemplifies this paradigm and at the end of the paper we ask further research questions for this direction.

1 Motivation: the Future is Green

As worldwide demand for energy increases, our present world realizes that energy resources are valuable assets, and researchers simultaneously aim to develop techniques to generate energy from renewable resources and to ensure efficient usage of energy so that energy derived notably from non-renewable energy resources is not unnecessarily wasted.

The ICT sector in 2008 contributed to 2% of global carbon emissions [5] i.e. 830 $MtCO_2e$, and expected to increase to 1.43 $GtCO_2e$ by 2020. In addition to making devices more energy-efficient thus reducing the carbon footprint, ICT stakeholders aim to utilize ICT to enable energy efficiency across the board in other non-ICT areas, in order to achieve energy savings of 15% (7.8 $GtCO_2e$) of global emissions by 2020. Thus, not only will ICT substantially influence global energy consumption, ICT mechanisms and networking will feature prominently in other non-ICT areas for better utilization of energy.

In the networking context, telecoms providers are moving to energy-efficient equipment and networks, and it is expected that by 2013 such equipment will be 46% of global network infrastructure [16]. For many years now, networking researchers have investigated ways to design and implement energy-efficient devices, networks and mechanisms ranging from lightweight resource-constrained devices like RFIDs and wireless sensor nodes to energy-aware routing algorithms.

* Post-proceedings printed on environmental-friendly acid-free paper to maximize durability and thus minimize printing effort and need for new paper; in line with our amortization of computations paradigm.

Within network security, researchers have investigated impacts on energy due to authentication and key exchange protocols [6, 7], notably for wireless sensor networks (WSNs) where prolonging battery life is of utmost importance. Researchers have also compared the energy consumption of different cryptographic mechanisms e.g. RSA and ElGamal implemented in WSNs [14], pairing based cryptography notably key exchange [22]; and shown that public-key cryptography has non-negligible impact on sensor lifetime [4].

Most of these work consider energy efficiency of *implementation* on specific hardware, or compare energy consumption of cryptographic primitives on certain platforms. Few have treated how to *design* a cryptographic scheme whose structure is well suited for energy efficiency. Indeed, cryptographic schemes are central to security protocols in different layers of the network protocol stack, e.g. SSL/TLS at the transport layer, IPsec at the network layer, so if one aims to design energy-efficient security protocols for these layers, it makes sense to ensure that the underlying cryptographic schemes used by these energy-efficient protocols are also designed with energy efficiency in mind. To the best of our knowledge, a couple of such results exist, as to be discussed next.

Indeed, the solution to the energy efficiency problem should be a holistic one e.g. considering all levels of abstraction, with no obvious weak link in the green sense.

Related Work. Kaps et al. [13] made recommendations for cryptographic design suited for ultra low-power settings:

* scalability: able to efficiently scale between bit serial and high parallelism so that implementers can trade off speed for power
* regularity: only a few different primitives should be used
* multihashing and multiencryption: sequentially calling a simpler and seemingly less secure hash function or encryption multiple times to achieve higher security. This approach is similar to the iterated structure of constructing hash functions based on compression functions and ciphers based on round functions
* precomputation/offline: the bulk of the processing is performed offline, before going online to process the incoming input so there is less latency, thus less energy wasted during the wait.

[13] also contrasted different alternatives to implementing basic functions e.g. algebraic representation versus table lookup, polynomial arithmetic for hardware implementation, constant shifts/rotations vs data-dependent ones, logic functions vs arithmetic ones. These are more in terms of choosing the proper basic functions to minimize energy consumption.

Meanwhile, Troutman and Rijmen [23] advocate a green approach to the design process, i.e. recycling primitives since long established ones garner more trust. They illustrated the concept with a discussion on AES' pedigree and how AES-type primitives were subsequently recycled in some manner in many SHA-3 candidates.

Rogaway and Steinberger [18, 19] show how to reuse block ciphers to construct hash compression functions, essentially emphasizing on the minimalist approach while at the

same time attaining assurance (given that the reused primitive has been in existence for some time and therefore seen considerable public analysis). The minimalist approach gives rise to less hardware or smaller memory footprint. More interestingly, in [19], they consider the approach of fixing the underlying block cipher's key input so that the corresponding block cipher key schedule can be computed offline instead of in online streaming mode, thus achieving better efficiency.

Essentially, the approach to recycle primitives advocated by [13, 23] is aimed to optimize the efforts (and energy) of designers that had already been spent on designing a primitive, and to optimize the code size of the primitive's implementation. Indeed, recycling a primitive means less energy needs to be spent to design a new one, and less code size is taken up by having multiple different primitives since the same primitive code can be called multiple times instead of different codes for different primitives.

The approach in [13] to perform the bulk precomputation minimizes online time (i.e. time when actual input data is incoming and needs to be processed). This approach is similar to concepts in cryptography relating to remotely keyed encryption [24] and online/offline signatures [20] for the application of real-time streaming. Less online time means less time is dependent on receiving the transmitted input which may cause latency and therefore unnecessary usage of energy during then.

Minimal Energy Consumption and Lightweightness vs Energy Efficiency. It should be noted that minimizing energy consumption or emphasis on lightweight design do not necessarily imply energy efficiency, and vice versa. By definition, energy efficiency refers to efficient use of energy, without unnecessary wastage. A lightweight cryptographic scheme is designed to use low-power computations, have small code size or require small memory space, yet it may involve multiple different low-power operations; so from the viewpoint of recycling primitives, this approach is not energy efficient. In contrast, a design that efficiently reuses primitives, may have higher power requirements than a lightweight scheme if the primitive itself involves complex operations that have high power consumption.

This Paper. Here, we first discuss the notion for green cryptographic engineering wherein are approaches for energy-minimizing and/or energy-efficient cryptographic processing. We also propose an approach such that the energy consumed in performing any substantial computation, is optimized by amortizing the output of the computation over different states of the scheme: we denote this as *amortization of computations*. We describe the relevance of this notion against recent cryptographic schemes proposed in literature. We conclude by posing further questions to be answered, some of which we are currently investigating.

2 Green Cryptographic Engineering Paradigm

Being green means being energy efficient, and we will use these terms interchangeably. Efficient refers to being fit for purpose and not being wasteful: both in the sense that no energy is drawn unnecessarily and that energy once drawn should be put to good use. Energy here is in the sense of any kind of resources. To appreciate this *resources* term, it is worth being explicit here what kinds of resources one may be interested in not wasting:

- ⋆ human effort:
  within a cryptographic engineering paradigm, we can think of making efficient use of the effort (and time spent is therefore also implied) of cryptographic designers, cryptanalysts, implementers and users of cryptographic schemes.
- ⋆ computational effort:
  efficient use of this resource can be in the sense of minimizing the amount of computation required (and therefore the cost of access to such computational power), or making full use of the output of any computation.
- ⋆ space:
  space refers to the capacity required to store or execute the scheme's implementations. This includes, e.g. ROM or RAM size especially for embedded computing platforms.
- ⋆ energy supply:
  computational machines require electrical power to run, so being green here could mean minimizing the amount of electrical power required by such computations.
- ⋆ time:
  the issue of interest here is to minimize the amount of time required to perform cryptographic operations. Towards that aim, researchers could investigate parallelizable structures that reduce the time-per-output ratio.

Cryptographic engineering, i.e. the process life cycle for design, analysis, implementation and use of cryptographic schemes and cryptographic based security systems, can be approached with a green strategy, bearing in mind the aim to optimize usage of the above listed resources.

Each stage of the process can be green in the following directions:

- minimized energy consumption:
  there is considerable research in this regard, notably the work on designing lightweight cryptographic schemes for resource-constrained devices such as RFID and wireless sensor networks; lightweight in the sense of energy consumption, code size and/or memory requirements.
  Applying this approach to cryptographic scheme design, lightweight and/or low-energy operations can be selected for use as basic building blocks within the cryptographic scheme, such as logical operations, and addition/rotation/XOR (ARX) constructions rather than multiplications.

- amortization of primitives:
  this approach is essentially in terms of recycling primitives that have already been designed or implemented [13]. This efficient usage of primitives leads to efficient code size, since the size remains the same irrespective of how many times the primitive is run. The regularity and multihashing/encryption suggestions of [13] are of this type of approach.

  This approach is implicit in typical cryptographic designs for efficiency and simplicity, e.g. iterative block cipher structures, feedback shift register based stream ciphers, Merkle-Damgård hash function structures, modular design paradigms i.e. constructions based on fundamental building blocks, and modes of operation that transform existing primitives into other primitives e.g. stream cipher, hash function's compression function or message authentication code from block cipher.

  This approach is also implicit in the cryptanalytic community, where the effort invested in discovering new cryptanalytic techniques is amortized via its application (at times via some adaptation or generalization) to multiple schemes of the same type or of different types, e.g. differential cryptanalysis and the notion of distinguishers initially invented for block ciphers, later applied to stream ciphers and hash functions.

  Towards a longer term aim, one can design fundamental operations that can form generic structures for use as building blocks within different types of cryptographic schemes, e.g. the inversion based Sbox of AES used to construct the AES round function, the LEX stream cipher and the AES-inspired SHA-3 candidates. Or design common primitives used for a multi-type cryptographic scheme, e.g. in the case of ARMADILLO [2].

- input-independent bulk (pre)computation:
  the approach here it to partition the computation effort into input-dependent and input-independent functions, and where the bulk of computations are designed into the input-independent functions so that most computations can be done in offline mode or so that this scales well even for inputs of large sizes or that are time-consuming, thus drawing less energy.

  This approach is exemplified in hash function designs e.g. Fugue [12]; where the security of such designs relies on heavy-weight finalization functions such that the iterated message block-dependent compression functions can be designed to be less computationally intensive.

- amortization of computations:
  the approach here is to reuse the effort put into a computation, more specifically, reflected in its output, multiple times at that state of output and in subsequent states in feedforward fashion. Doing so allows subsequent states to be more directly influenced by the current state 'for free', i.e. without extra computations to produce that state;

and the net effect of this is that we then require less number of iterations overall in order to retain similar level of security.

This approach is exemplified in the structure we propose later in this paper. Our structure is a generic one that subsumes feedforward-based constructions in literature, e.g. block cipher based hash compression functions like Davies-Meyer and Miyaguchi-Preneel are special cases when the state is only reused once, as well as more recent schemes that we will discuss in more detail in section 4.

During the conference, we hope to engage attendees in a discussion of other possible green approaches to the cryptographic engineering process, or whether there are any other instantiations of the above-listed approaches in existing cryptographic schemes.

3 A Computation-Amortizing Structure

One energy-efficient cryptographic structure that instantiates the amortization of computations approach is as follows. As with conventional design strategy, the structure is iterative, based on a round function $F(s_i, s^*_{i\_})$ where s_i denotes the conventional type of input state to the function and where $s^*_{i\_}$ denotes one or more previous states for $i\_ < i$.

Within $F()$, we have the following steps in sequence:

1. $s_i \leftarrow \mathsf{heavy}(s_i)$
2. $s_{i+1} \leftarrow \mathsf{light}(s_i, s^*_{i\_})$

where $\mathsf{heavy}()$ is a computationally-intensive operation e.g. multiplication, while $\mathsf{light}()$ is a lightweight operation e.g. logical functions.

This way, from the computational viewpoint, the input vector $s^*_{i\_}$ does not substantially add to computational requirements (and thus energy) of F, and yet from a cryptographic viewpoint, adds substantially to the mixing process within F.

The basic idea here is that an intermediate state s_i after computation function F, is reused several times at the same state location (i.e. spatial amortization) and also fedforward in time (i.e. temporal amortization) to be reused in subsequent states s_j $(j > i)$, and aside from the first time that the state is used, subsequent uses of that state will involve non-computationally intensive functions to mix that state back in. The gist is to fully utilize (and reuse) any state including the outputs of any function, as many times as possible; essentially using the 'copy' and 'feedforward' (these are computationally non-intensive) operations.

Amortization then comes from the fact that the light inputs are actually the reusing of states from other parts of the structure (so we kind of have them for free without having to do extra computations to get them), and the way that they should influence the F function should be in a non-computationally intensive manner e.g. XOR, logical operations.

In this way, it is intuitive that the required number of rounds can be less than conventional structures of F that process only the current state s_i, while maintaining the security level.

In analyzing this kind of construction, a new measure so-called *points of influence* (POI) can be considered. This notion measures how many other state points are immediately and directly affected by a change in a state at one point within a cryptographic scheme. Indeed, the probability of a differential distinguisher should be reduced given a higher points of influence, because the probability is a function of the number of active points affected by a difference in a state.

A preliminary step concerning diffusion when looking at F as a round function could be to consider that the injection of the $s^*_{i\_}$ state looks like a subkey addition in the block cipher context or a message reinjection in the hash compression function context. Indeed, it turns out that block cipher key addition and hash function message injection operations are typically designed to be light, thus fitting nicely with this convention. Thus, a first insight concerning diffusion could be to look at diffusion properties of the key into the cipher. A parallel could be made to evaluate the diffusion when considering $s^*_{i\_}$ taking into account the points of influence. Concerning hash functions, the theory of goal would be to readapt security proofs to the case of several intermediate message dependencies.

From this notion of diffusion which is among the most important when talking about cryptography (the other one is of course the confusion), we could derive the probability of success when looking at an adversary within the context of the differential or the linear distinguishers.

4 Computation Amortization in Practice

Our computation-amortizing structure in section 3 can in fact fit different kinds of cryptographic schemes in literature, including hash functions, message authentication codes (MACs) and stream ciphers.

For hash function structures, the feedforward strategy, i.e. to feed a state (output from some function) forward to be combined with a future state, is well established. For instance, the PGV [17] structure for constructing hash compression functions from block ciphers is a popular example of this strategy in practice.

This strategy is also exemplified in recent hash function structures constructed from compression functions where the feedforward enters two (instead of just one) future states. These include the 3C and 3C+ [9] constructions, where up to two copies of each state are fedforward to the final state for combination; and the ESS [15] and its predecessor [21], where one of the states is forwarded for combination with two different states.

More interestingly, related compression function constructions appear in independent work due to Rogaway and Steinberger in [18, 19]. For some input x, the function of [18] is

defined as:

$$
\begin{aligned}
&\text{for } i \leftarrow 1 \text{ to } k \text{ do} \\
&\quad s_i \leftarrow f_i(x, s_1, \ldots, s_{i-1}) \\
&\quad y_i \leftarrow \pi_i(s_i) \\
&\text{endfor} \\
&\text{return } g(x, s_1, \ldots, s_k)
\end{aligned}
$$

where f_i denotes some function and π_i denotes some permutation. If we let the f_i and g functions be lightweight functions, and indeed, the particular compression function construction in [19] is one where such f_i and g are linear functions, i.e.

$$f_i = a_0 x + \sum_{j=1}^{i-1} a_j s_j,$$

where a_j (for $j \in \{0, \ldots, i-1\}$) is some constant and we let $g = f_{k+1}$; then the resultant construction of [19] can be seen as an instance of our computation-amortizing structure of section 3, where our heavy() function is instantiated with a permutation π_i.

In terms of MAC structures, the SS-NMAC scheme of [8] builds on the structure of [21] and retains the feedforward strategy. More specifically, for about half of its internal functions, the output state is fedforward onto two other states for combination.

For stream ciphers which are traditionally represented by first an updating function f that updates the content of an internal state s_i and then a filtering function g that filters the content of s_i, the model proposed for computation-amortizing structure leads to intrinsically modify the stream cipher in the following way:

- ∇ If the updating function f takes the role of the computation-amortizing F function described in section 3 to act on s_i and on s^*_{i-}, then this means that the internal state becomes bigger. By this way maybe the f function could be lightened due to its bigger internal state. An example of a stream cipher that has a memory state is given in [3] for software cases. The case where g the filtering function takes the role of the F function must be carefully studied because at this time, it is not so clear that adding a light() part could be so directly derived to discard classical attacks because the resistance of a stream cipher against traditional distinguishers mainly depends on the clever choice of g.

Another way to build stream ciphers is to use a block cipher in the so-called counter mode (CTR) of operation where the keystream is generated by encrypting successive values of a counter, and then each plaintext block is XORed with each keystream block. As it is not possible to directly modify a block cipher using the computation-amortizing structure proposed in section 3 (due to the required invertibility property for conventional block

cipher structures), we could try to modify the counter mode itself reinjecting previous values. For example, we could derive those possible modes from MILENAGE, the one-block-to-many mode used in 3GPP [1] which was proved to be secure in [10].

5 Open Research Questions

While it may be that no matter how energy *in*efficient the cryptography is and still it is unlikely to be the most inefficient component of a system, yet the right strategy should be to approach this analogous to security's celebrated weakest link property, i.e. viewing a system to be only as energy-efficient as its most energy-inefficient component. Thus, as researchers work hard to design and implement network security systems and protocols that are energy-efficient, the cryptographic schemes that underlie these network security protocols should also be designed with explicit energy efficiency.

The green cryptographic engineering paradigm discussed in section 2 is a good start towards approaching this aim, so that the process of engineering cryptographic schemes (design, analysis, and/or implementation) can be performed while maintaining energy efficiency.

The motivation is clear. Security is already largely seen frequently as an impediment to performance, yet needs to be there in view of constant threats of attacks. In a future where energy resources are increasingly becoming a scarcity, security will become all the more undesirable if in addition to trading off performance, they are also energy *in*efficient.

We conclude for now with some further research questions for this particular direction:

† Can we design cryptographic structures that are provably secure and provably energy efficient?

† Can we transform provably secure cryptographic structures into ones that are energy efficient, while still retaining provable security?

† How do we model energy efficiency in the provable sense?
To this aim, we have started to formalize faulty and/or loss of energy models where the designer or the user is not malicious in the security sense but does not take any care towards the energy efficiency goal (because he could be lazy, ignorant or indifferent to energy efficiency). Then provable energy efficiency for a scheme is to show that energy loss only occurs with negligible probability.

† What provably energy efficient notions can be defined?

† What metrics can be used to measure energy efficiency of cryptographic schemes?
Along the lines of the green cryptographic engineering paradigm approaches discussed in section 2, we suggest investigating metrics such as *primitive multiplicity* (the number of times a primitive is recycled), *state multiplicity* (the number of times a state is used elsewhere), *points of influence* (how many other state points are immediately affected when a change is made in one state).

References

1. 3rd Generation Partnership Project, "Specification of the MILENAGE Algorithm Set: An Example Algorithm Set for the 3GPP Authentication and Key Generation Functions f1, f1*, f2, f3, f4, f5 and f5* - Document 2 (TS 35.206): Algorithm Specification ; Document 5 (TR 35.909): Summary and Results of Design and Evaluation". Available online at `http://www.3gpp.org`
2. S. Badel, N. Dagtekin, J. Nakahara, K. Ouafi, N. Reffé, P. Sepehrdad, P. Susil and S. Vaudenay, "ARMADILLO: A Multi-purpose Cryptographic Primitive Dedicated to Hardware," *Proc. CHES '10*, LNCS 6225, 2010, pp. 398-412.
3. T.P. Berger, M. Minier and B. Pousse, "Software Oriented Stream Ciphers Based upon FCSRs in Diversified Mode," *Proc. INDOCRYPT '09*, LNCS 5922, 2009, pp. 119-135.
4. K. Bicakci, H. Gultekin and B. Tavli, "The Impact of One-Time Energy Costs on Network Lifetime in Wireless Sensor Networks," *IEEE Communications Letters*, Vol. 13, No. 12, 2009, pp. 905-907.
5. The Climate Group, "SMART 2020: Enabling the Low Carbon Economy in the Information Age," *Global e-Sustainability Initiative (GeSI)*, 2008.
6. G. de Meulenaer, F. Gosset, F.-X. Standaert and O. Pereira, "On the Energy Cost of Communication and Cryptography in Wireless Sensor Networks," *Proc. WiMob '08*, IEEE, 2008, pp. 580-585.
7. O. Delgado-Mohatar, J.M. Sierra, L. Brankovic and A. Fuster-Sabater, "An Energy-Efficient Symmetric Cryptography based Authentication Scheme for Wireless Sensor Networks," *Proc. WISTP '10*, LNCS 6033, 2010, pp. 332-339.
8. Y. Dodis and J.P. Steinberger, "Message Authentication Codes from Unpredictable Block Ciphers," *Advances in Cryptology - CRYPTO '09*, LNCS 5677, 2009, pp. 267-285.
9. P. Gauravaram, W. Millan, E. Dawson and K. Viswanathan, "Constructing Secure Hash Functions by Enhancing Merkle-Damgård Construction," *Proc. ACISP '06*, LNCS 4058, 2006, pp. 407-470.
10. H. Gilbert, "The Security of "One-Block-to-Many" Modes of Operation", *Proc. FSE '03*, LNCS 2887, 2003, pp. 376-395.
11. L. Greenemeier, "Can the World's Telecoms Slash their Energy Consumption 1000-Fold?," *Scientific American*, 11 January 2010.
12. S. Halevi, W.E. Hall and C.S. Jutla, "The Hash Function "Fugue"," *SHA-3 Candidate Submission*, 15 September 2009.
13. J.-P. Kaps, G. Gaubatz and B. Sunar, "Cryptography on a Speck of Dust," *IEEE Computers*, Vol. 40, No. 2, 2007, pp. 38-44.
14. R. Kayalvizhi, M. Vijayalakshmi and V. Vaidehi, "Energy Analysis of RSA and ELGAMAL Algorithms for Wireless Sensor Networks," *Proc. CNSA '10*, CCIS 89, 2010, pp. 172-180.
15. A. Lehmann and S. Tessaro, "A Modular Design for Hash Functions: Towards Making the Mix-Compress-Mix Approach Practical," *Advances in Cryptology - ASIACRYPT '09*, LNCS 5912, 2009, pp. 364-381.
16. Pike Research, "'Green' Telecom Equipment will Represent 46% of Network Capital Expenditures by 2013," *Pike Research*, 9 June 2009.
17. B. Preneel, R. Govaerts and J. Vandewalle, "Hash Functions based on Block Ciphers: a Synthetic Approach," *Advances in Cryptology - CRYPTO '93*, LNCS 773, 1993, pp. 368-378.
18. P. Rogaway and J. Steinberger, "Security/efficiency Tradeoffs for Permutation-based Hashing," *Advances in Cryptology - EUROCRYPT '08*, LNCS 4965, 2008, pp. 220-236.
19. P. Rogaway and J. Steinberger, "Constructing Cryptographic Hash Functions from Fixed-key Blockciphers," *Advances in Cryptology - CRYPTO '08*, LNCS 5157, 2008, pp. 433-450.
20. A. Shamir and Y. Tauman, "Improved Online/Offline Signature Schemes," *Advances in Cryptology - CRYPTO '01*, LNCS 2139, 2001, pp. 355-367.
21. T. Shrimpton and M. Stam, "Building a Collision-Resistant Compression Function from Non-compressing Primitives," *Proc. ICALP '07*, LNCS 5126, 2008, pp. 643-654.

22. P. Szczechowiak, A. Kargi, M. Scott and M. Collier, "On the Application of Pairing based Cryptography to Wireless Sensor Networks," *Proc. WiSec '09*, ACM, 2009, pp. 1-12.
23. J. Troutman and V. Rijmen, "Green Cryptography: Cleaner Engineering through Recycling," *IEEE Security and Privacy*, Vol. 7, No. 4, 2009, pp. 71-73.
24. R. Weis, W. Effelsberg and S. Lucks, "Remotely Keyed Encryption with Java Cards: a Secure and Efficient Method to Encrypt Multimedia Streams," *Proc. ICME '00*, IEEE, 2000, pp. 537-540.