Providing security in energy harvesting sensor networks

Sylvain Pelissier, Prabhakar T.V. , H.S. Jamadagni, R. VenkateshaPrasad, Ignas Niemegeers
Centre for Electronics Design and Technology
Indian Institute of Science
Bangalore, India
sylvain.pelissier@epfl.ch, (tvprabs,hsjam)@cedt.iisc.ernet.in,r.r.venkateshaprasad,i.g.m.m.niemegeers@tudelft.nl

***Abstract*—In this work, we study the adaptability of well known cryptography algorithms to energy harvesting wireless sensor networks. We are particularly interested in algorithms that have the ability to adapt to varying power in such networks. Our investigations and implementation on hardware platforms indicate that it is optimal to precompute a few key stream bytes, store in memory and later used during the time when the system is low on harvested power level. Our demonstrable setup shows using a precomputed key stream can decrease the energy consumption by 14%. We have implemented the Trivium stream cipher for two different microcontrollers, the MSP430 and the AVR ATmega128l and show the performance results for these implementations. We have implemented an algorithm based on universal hash functions to provide message authentication with the assistance of stream ciphers. We show that this authentication algorithm has exciting properties for energy harvesting system and more generally for resource constrained devices.**

***Index Terms*—Security, Energy harvesting, Cryptography, Wireless sensor network, Stream cipher, Universal hashing, Wegman-Carter authenticator, Trivium, Poly32, AVR, MSP430, ATmega128.**

I. Motivation and related work

The world energy consumption reduction and the urgent need to conserve natural resources has inspired users towards energy usage reduction and efficiency. Perhaps homes and buildings are one of the higher side consumers of the world energy and an intervention to reduce this consumption has a significant impact. This is because most energy requirements are human presence centric. For instance, building heating and cooling is required when humans are present. Similarly, lighting the office, home and corridor spaces require significant energy when humans are present. With the advent of Information Communication Technologies and specifically wireless sensor network technologies (WSN), energy consumption reduction is possible by deploying presence detection and other sensors. For example, the home and office AC may be powered on based on multiple parameters such as the presence, ambient temperature and humidity. Thus, based on movement or detecting the presence of a specific person inside a home, several home and office appliances may be powered on. It is estimated that nearly 4.5% of global energy savings is possible with such Information and Communication Technologies (ICT) with nearly 8.5% reduction in carbon-dioxide [1]. Every future home and office device will become a wireless networked "smart object" with appropriately equipped sensors possessing a short range communication capability using devices such as IEEE 802.15.4 radio transceivers.

There are however, two major hindrances for such sensor and communication network percolation in our daily lives. Firstly, each of these devices need a power source such as a battery to power the sensor node. This would also mean that battery replacements have to be done periodically with the risk of partitioning or shutting down the network temporarily. It is estimated that the share of ICT in the worldwide electricity consumption will grow further in the foreseeable future and the present trends indicate this share anywhere between 11% and 20% by the year 2020 [2]. Secondly, providing security in these environments becomes a prime requirement. For example, authentication is important in home and office environments. Similarly, confidentiality of data is of outmost importance in health care applications such as in-home patient monitoring. A patient's blood pressure and heart rate, for example could be measured by wearable devices capable of wirelessly communicating the data to a home computer running a pre-screening analysis software [3]. Thus with an expected thousand ICT devices per person in 2017 [4], the problem of powering these devices together with secure communication is in itself a significant task. We think that energy harvesting for home and office environments sufficient to drive wireless sensor networks is a viable option. For instance, a palm sized solar panel can already generate sufficient energy (about 300 mW of power) under medium to bright sunlight. Similarly, the inside room heating and outside of buildings is a sufficient temperature difference required for most thermoenergy generator modules. While cryptographic algorithm complexity, strengths and vulnerabilities are well researched areas, little is known in literature about their suitability to energy harvested wireless sensor networks. Therefore, the question that remains is about the possibility of a fully secure communication with energy harvested sensors. This paper addresses this key question. Our sensor nodes are driven using solar energy simulators and our studies are restricted to providing security under a fluctuating power source.

In the energy harvesting paradigm, the lifetime of a network can be considered as infinite (besides hardware failures) but the power that was considered as constant in battery powered sensor networks can have spatio-temporal variation over the

complete network. While all existing algorithms including cryptographic algorithms adapted to WSNs have the goal to reduce the energy consumed and thus to maximise the lifetime of the networks, in the paradigm of energy harvesting sensor networks, the algorithms need to be energy-neutral *i.e.,* each operation needs to consume less or equal energy to that of the energy harvested. In this context, new algorithms have been developed for power management [5], routing [6] and other networking operations.

Several investigations on block ciphers are reported for wireless sensor networks. Dilda *et al.* [7] implemented AES (Advanced Encryption Standard) for Texas Instrument's (TI) low power microcontroller (MCU) MSP430 in C language. Their implementations reach a data rate of 286.35 kb/s when MSP430 uses a clock frequency of 8MHz, resulting in 224 cycles/byte. Their implementation requires 5160 bytes in Flash memory and 260 bytes in SRAM. This implementation was perhaps the first one over MSP430 MCU that matches the 250 kb/s rate requirements of the IEEE 802.15.4 radio. We are not aware of any assembly implementation results for AES for this platform.

On Atmel's AVR 8-bit microcontrollers, The fastest AES implementation on AVR was reported by Osvik *et al.* [8]. Their assembly implementation of AES has an encryption rate of 125 cycles/byte and a decryption rate of 181 cycles/byte. Their implementation consumes 1912 bytes in Flash memory and 176 bytes in SRAM.

The consumption of block cipher modes of operation in WSN has been studied by Bauer *et al.* [9]. They studied four different authenticated encryption with associated data (AEAD) schemes on crossbow's MICAz mote. They show that CCFB+H is the most suitable for this platform and that the performance of the GCM mode is significantly behind the other. Huang *et al.* [10] implemented the NH function family of the UMAC algorithm on the MSP430 MCU. This implementation shows a good performance in terms of computation. However, the tag length is 32 bits whereas the function itself is 2^{-16} almost universal so the MAC security is only 16-bit. This implies that half of the bits of the tag are wasted. Since the communication is far more energy consuming than computation in a WSN, this solution is not suitable to constrained devices.

Stream ciphers have been also studied in WSNs. A benchmark of the profile I ciphers in the eSTREAM portfolio was implemented on the 8-bit microcontroller called the MicaZ sensor node by Meiser *et al.* [11]. They show that each stream cipher except Salsa20 has better performance than AES in CTR mode.

The goal of this work is to investigate suitability of cryptographic algorithms and their adaptability to the special paradigm of energy harvesting sensor networks. We study their behavior by an implementation. We only use well-studied algorithms and not create our own algorithms which could address the new issues of energy harvesting sensor networks. This approach allows avoiding the construction of efficient but totally insecure algorithms. We constructed solutions which are completely modular. Each building block of our solution can be replaced by another which offers the same features but with other characteristics. This approach seems to us the best since the applications of WSNs are so vast that the constraints from one case to another can be very different. We investigated the optimization of the algorithms that are widely used in order to save as much energy as possible. Finally, to illustrate our proposal, we provide performance results for each implemented algorithm on the selected platforms as well as a practical demonstration.

II. Algorithms for energy harvesting sensor networks

In this section we show that buffering key stream bytes from a stream cipher during times of high energy is optimal. We also propose a method to adjust the buffer size. We show message authentication code tags derived using universal hashing is also optimal for energy harvesting sensor networks.

A. Stream ciphers

Stream ciphers are designed to be less energy consuming than block ciphers. In addition, the generated key stream bytes are then exclusive OR operated (XOR) with the plain text bytes. It means that the key stream bytes can be computed without the plain text. Our idea was to compute key stream bytes as a function of the power available and store it. If the power is low, the key stream bytes are only read from memory and XORed with the plain text. If the power is high, the next key stream bytes can be precomputed and stored again. The stream cipher algorithm can be implemented in software or the key stream can be generated from special hardware and stored in memory. Some IEEE 802.15.4 transceivers like the Texas Instrument's (TI) CC2420 offer a hardware implementation of AES in counter mode. This implementation can be used to generate the key stream. However we studied only the capability of software implementations.

Trivium is a stream cipher proposed by De Canière and Preneel [12]. This cipher was designed to be efficient when it is implemented in hardware and thus was proposed as a Profile II candidate of the eSTREAM project. We selected this stream cipher and implemented it in software. The software implementation of this cipher can be very flexible. For instance, we can choose to take advantage of speed and implement a version which output 64 bits of the key stream at each iteration or we can privilege to reduce the memory footprint and output only one word at each iteration. We found this property suitable for constrained devices so we chose to implement this stream cipher.

B. Wegman-Carter authenticators

The concept of universal hashing was introduced by Carter and Wegman [13]. Their theoretical work allows constructing message authentication code which are provably secure and often with better performances than other message authentication code algorithms. In addition this is the

only general and secure construction which allows providing authentication with a stream cipher. The theory of universal hashing is based on universal functions. Generally speaking, a universal hash function is a function which maps a message of any length to a fixed length hash value and the probability that two messages have the same hash value is very low.

We chose to implement an universal hash function family: the Poly32 [14] function family. The Poly32 family has a very small key size of only 32 bits. It allows having small memory consumption but the collision probability increases with the size of the message. In the context of a WSN this drawback is not a big flaw. For example, if the specifications of IEEE 802.15.4 are applied, a message payload is at most 103 bytes and we authenticate less than 128 bytes. If we authenticate each IEEE 802.15.4 frame then the forgery probability of Poly32 is at most $32 \cdot 2^{-28} = 2^{-23}$ for a 128 byte packet. In certain situations, it can be enough but in other it is clearly sufficient. To have a higher security level, the sensor nodes can compute the algorithm twice with two different keys. This ensures the collision is squared but then the communication cost and the key length are twice bigger.

C. Analysis

Since we have seen that buffering the key stream is probably an efficient method when using stream cipher, in the context of energy harvesting system, we now provide a theorem that describes the conditions for feasibility of the buffered stream cipher method. We also provide a method to adjust the size of the buffer.

To compute the size of the buffer, we consider a basic model where a power management system inputs to the sensor node, a duty cycle $D(i)$ at every time period i of duration T. During a period i, the mote is active for a time $\tau(i) = D(i) \cdot T$. We assume that if $\tau(i)$ is less than a threshold a, then the mote can do its operation (transmission and computation). It does not however have the time to compute the key stream bytes to fill up the buffer. Above this threshold, the sensor node can fill the buffer during a period $\tau_f \leq \tau(i) - a$. We assume that during the active period, the sensor node transmits at a rate of r bytes of encrypted data per second. We also consider b_f the number of bytes per second the sensor node can fill the buffer. Basically, a sensor node can use a buffered stream cipher if for each period, it reads fewer number of bytes from the buffer compared to the actual buffer size and the number of key stream bytes that can be produced. Theorem 1 gives a condition to achieve such a system.

Theorem 1: A system stores and reads a key stream from a buffer of size B with an initial level $L(0)$ can work if and only if

$$L(0) + \sum_{j=0}^{i} (\max(\tau(j) - a, 0) \cdot b_f - \tau(j) * r) \geq 0$$

for all integer i.

Proof: For a period i, the sensor node reads $\tau(i) * r$ bytes from the buffer in order to transmit its packets. The buffer will be filled by at most $\max(\tau(i) - a) \cdot b_f$ bytes. Let us consider the function L which takes as parameter a period i and outputs the maximal level of the buffer after this period. $L(0)$ is the size of the buffer before the mote starts working. Then we have that

$$L(i+1) = L(i) + \max(\tau(i) - a) \cdot b_f - \tau(i) * r$$

or without recursion

$$L(i+1) = L(0) + \sum_{j=0}^{i} (\max(\tau(j) - a) \cdot b_f - \tau(j)) * r)$$

We see that the system works properly if and only if for each period i the size of the buffer is non-negative *i.e.,* if and only if $L(i) \geq 0$ for all $i \geq 0$. ■

This theorem, does not give a way to compute the size of the buffer but it allows checking if a system will work properly or not. We can also see that without a buffer, the condition becomes $(\max(\tau(j) - a, 0) \cdot b_f - \tau(j) * r) \geq 0$ for all integer i. It shows again that this method is applicable in many more situations. In addition, the theorem indicates the conditions under which it is useful to apply the buffered key stream method. If the function L is an increasing function, then it is not useful to apply this method since the key stream used can be generated during the same period. If L is a decreasing function, then the system is not well defined and cannot work at that duty cycle.

We provide a simple model that allows us to compute the size of the buffer. We suppose that τ is a periodic function of period T_τ. During this period, the mote is able to store the key stream only during a time τ_f at the beginning of the period. Then the system works properly if the key stream generated is equal to the key stream consumed during one period T_τ:

$$\tau_c b_f = T_\tau r$$

Since we want to store in the buffer all the key stream bytes generated, we have:

$$B = \tau_c b_f$$

The result is quite apparent. It means that at each period, we store the number of bytes we will need to use. let us consider a practical example. Consider a stream cipher with $b_f = 25000$ that is almost the rate of Trivium on MSP430. We also assume that $r = 300$ and $T_\tau = 5$ seconds. The sensor node must be allowed to compute the key stream during $\tau_c = 60$ ms period and thus must have a buffer of size $B = 1500$ bytes.

The previous model can fit some practical sources that are periodic. For more complex model, the size of the buffer can be found by linear programming since we need to find the minimum buffer size under the condition given by the previous theorem.

III. Implementation results

A. Stream cipher implementation - Trivium

We have an inline assembly implementation of Trivium optimized for the ATmega128L microcontroller and a C implementation for the MSP430 microcontroller. For ATmega128L,

compilation was done with the avr-gcc compiler with -O2 and -Os optimization options. The cycle consumption was obtained with simulation in Avrora software. For MSP430, we used the IAR Workbench compiler and simulator. We show the performance results of our implementations in Table I. Table I

	Initialisation	Keystream generation (1 byte)
ATmega128L	44300	153
MSP430	24769	160

TABLE I
COMPUTATION CYCLES OF TRIVIUM

shows that the assembly implementation of AES by Osvik *et al.* [8] has better performances on AVR MCU with a higher security level. Therefore, AES-CTR can be a better choice for this platform if the speed is the main concern. However, Trivium has the advantage to be very flexible. It is trivial to adapt the implementation to the constraints specific to the platform and made another implementation on ATmega128L of Trivium with a ROM consumption of 970 bytes in Flash memory. On MSP430, we can notice that Trivium has better performances than AES implementation. It also has a very low energy and Flash memory consumption on this platform. This ensures Trivium is a well adapted cipher for WSNs.

B. Message authentication code Poly32 implementation

We also implemented the Poly32 function on the ATmega128l and MSP430 MCUs. We made measurements of cycle consumption for these hash function families on messages of sizes 9, 25, 37 and 128 bytes. The three first sizes are the same as the one used by Bauer *et al.* [9] in their measurements of AEAD schemes. We give Table II the results we obtained for our Poly32 implementations on these MCUs. Table II shows that this hash function is very efficient. On

Message size (bytes)	ATmega128L	MSP430
9	3929	594
25	7700	1300
37	12718	1827
128	40359	5695

TABLE II
COMPUTATION CYCLES OF POLY32

MSP430, the computations for this algorithm is six times lower then the ATmega128L results. This is because the algorithm is designed for 32-bit processors. Therefore, the 16-bit MSP430 is more suitable for this algorithm than the 8-bit AVR MCU. Moreover, the MSP430 has a hardware 16-bit multiplier that can speed up multiplications significantly.

On AVR platform, we implemented a Wegman-Carter authentication mechanism based on Poly32 and Trivium. We compared our results with the results of Bauer *et al.* . Fig 1 shows the results of the comparison.

Fig1 shows that if a 32-bit tag is required, then the Poly32-Trivium method is the fastest on this platform. However, if a

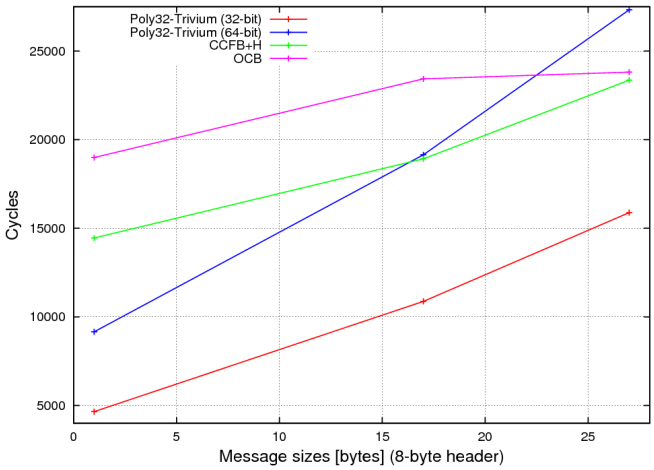

Fig. 1. Wegman-Carter authentication mechanism

64-bit tag is required then this method is faster only for small messages.

IV. PRACTICAL DEMONSTRATION

To show the gain of the buffered key stream method in practice, we setup a small demonstration with an energy harvesting system. The experimental setup consists of two sensor nodes. One is the transmitter running on harvested energy and the other the receiver working similar to a base station connected to a PC. The receiver counts the total number of messages received. The transmitter node is powered by an energy harvesting system connected to two solar panels. Above the solar panels there are four lamps of 20W each. The energy harvesting light source system was developed locally. For energy storage, we used a 30milliFarad ultra capacitor.

In this experiment, we first turn on the four lamps to allow charging the supercapacitors. The sensor node does not operate and start any communication activity until the supercapacitors are completely charged. Once the supercapacitors are charged, the initialisation algorithm is executed and then the sensor node enters into low power mode for a few seconds. Meanwhile, we turn off all the lamps so that the mote is only powered by the supercapacitors. Subsequently, the sensor node wakes up and starts sending packets with 8-byte encrypted payloads. We considered two different scenarios. In both cases, the Trivium algorithm is used for encryption. In one case the key stream is generated just before the packet is sent. In the other case, the key stream is computed during the first phase when lamps are switched on and stored in SRAM. The size of the buffer was derived with the formula given in Section II-C. Fig 2 illustrates the obtained results. Since the receiver counts the number of packet received we were able to compare the number of messages that can be sent between the two scenarios. We consider this demonstration as a sufficiently accurate measurement, unlike a measurement using a battery. This is

because issues such as state of charge, discharge characteristic curves are no longer applicable for super capacitors. Fig 2 does show some variations in the number of packets received. This is because of variations in the supercapacitor's terminal voltage due to slight timing mismatch during turning off the lamps. The results presented are an average of several attempts of the experiment.

When the key stream is not buffered, the sensor node

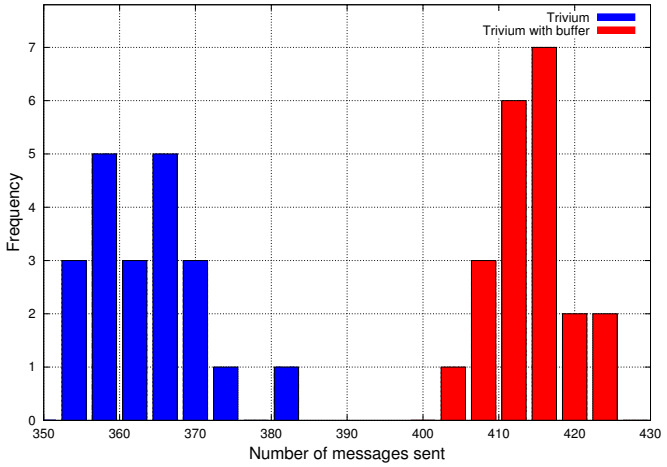

Fig. 2. Demonstration results

is able to send an average of 364 packets, whereas the in the second case, when the key stream is buffered, the sensor node can send an average of 415 packets. One may now conclude that on an average the sensor node can send 14% more packets when the key stream is buffered. This demo shows that for a practical energy harvesting system, it can help the sensor node during low power harvesting period where precomputed key streams are stored in memory.

V. Conclusion and further questions

We discovered that stream ciphers have the property that their key stream is independent of the plain text. This property allows an energy harvesting system to precompute the key stream bytes and store them in SRAM until used.

We chose to implement the Trivium cipher which has properties suitable for WSNs. We optimized its implementation and coded an assembly implementation of Trivium for the AVR platforms. We also implemented a universal hash function family. We provided performance results of our implementation on the ATmega128L and MSP430 MCUs. To the best of our knowledge, these algorithms have never been studied on small and constrained platforms. Our results show that Trivium has the same performance as other ciphers on the AVR MCU and has better performance than AES implementation on the MSP430 MCU. We have also remarked that universal hash functions have very efficient construction to provide authentication with stream ciphers. We derived some constraints for applying the buffered key stream method to energy harvesting sensor networks and we suggested a simple method to determine the buffer size. Finally, our experimental setup displays the benefits of using this method when the energy source is a photonic source and we measured a gain of 14% with our proposed scheme. This showed that indeed the method of storing some key stream bytes in memory when the power is high and to use them without computation when the power is low is practical with an increase in performance.

We have however not studied key agreement schemes and their necessary modification to adapt to power fluctuation. This could be considered for future work. We have also not studied networking issues with harvested energies. For instance, how the neighbor nodes can assist an energy constrained node for performing heavy computations. This approach, although looks attractive, can be less useful since the cost of communication is an order of magnitude higher than the cost of computation. Nevertheless working in these directions could lead to interesting developments.

VI. Acknowledgement

The fourth and fifth authors would like to thank IOP GenCom under *Future Home Networks* project funded by the Dutch Ministry of Economic Affairs.

References

[1] *h*ttp://www.marketresearch.com/product/display.asp?productid=2427947
[2] *E*nergy Consumption of Information and Communication Technology (ICT) in Germany up to 2010 http://www.isi.fraunhofer.de/e/eng/publikation/online/iuk/Fraunhofer-ICT-report.pdf
[3] S.Dagtas et al., *Multi-stage Real Time Health Monitoring via ZigBee in Smart Homes* In Proceedings of 2007 IEEE International Conference on Advanced Networking and Applications Workshops (AINAW), pp. 782-786
[4] N.Jefferies., *Global Vision for a Wireless World* 18th meeting of WWRF, June 2007, Helsinki, Finland
[5] A. Kansal, J. Hsu, S. Zahedi, and M.B. Srivastava. *Power management in energy harvesting sensor network.* In ACM Transactions on Embedded Computing Systems (TECS), volume 6, 2007.
[6] E. Lattanzia, E. Reginia, A. Acquavivaa, and A. Bogliolo. *Energetic sustainability of routing algorithms for energy-harvesting wireless sensor networks.* Computer Communications, 30:2976-2986, 2007.
[7] S. Didla, A. Ault, S. Bagchi *Optimizing AES for embedded devices and wireless sensor networks* Proceedings of the 4th International Conference on Testbeds and research infrastructures for the development of networks & communities, 2008.
[8] D. A. Osvik, J. W. Bos, D. Stefan, D. Canright *Fast Software AES Encryption* Fast Software Encryption, 2010.
[9] G.R. Bauer, P. Potisk and S. Tillich *Comparing Block Cipher Modes of Operation on MICAz Sensor Nodes* Proceedings of the 2009 17th Euromicro International Conference on Parallel, Distributed and Network-based, 2009.
[10] A.L. Huang *UMAC for Ultra-low Power Environments* Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2006.
[11] G. Meiser, T. Eisenbarth, K. Lemke-Rust and C. Paar *Efficient implementation of eSTREAM ciphers on 8-bit AVR microcontrollers* IEEE Third International Symposium on Industrial Embedded Systems (SIES), 2008.
[12] C. De Cannière and B. Preneel *TRIVIUM specifications* 2005. http://www.ecrypt.eu.org/stream/ciphers/trivium/trivium.pdf.
[13] J.L. Carter and M.N. Wegman *Universal classes of hash functions* Proceedings of the ninth annual ACM symposium on Theory of computing, 1977.
[14] T. Krovetz and P. Rogaway *Fast Universal Hashing with Small Keys and No Preprocessing: The **PolyR** Construction* Internatinal Conference on Information Security and Cryptology (ICISC) 2000.