MIT

6.857 Final Paper

Computer Networks Security

The Internet Of Insecure Things

Analyzing a low-energy protocol and cryptographic solutions

Authors:	*Email Address*
Lucas Camelo Sá	jlucas16@mit.edu
Amy Greene	greenea@mit.edu
James Loving	jloving@mit.edu
Ulziibayar Otgonbaatar	ulziibay@mit.edu

May 13, 2015

Abstract

For our 6.857 final project, we analyzed and suggested improvements to the ANT+ wireless communication protocol for embedded and wearable devices. We present our analysis of the ANT+ protocol, in which we found some provisions for encryption but none for cryptographic authentication. We researched potential low-energy MACs that could be used to augment the ANT+ protocol, and implemented them to compare their energy consumption. Based on our findings, a CMAC scheme with a light-weight block cipher like Simon will provide message integrity at a low energy cost, thus improving ANT+ protocol in terms of its security.

1 Introduction

The Internet of Things (IoT) is a fast-emerging ecosystem of Internet-connected devices that is changing the way society functions. IoT devices can be used to replace door-locks, to sense forest fires and landslides, to or to detect enemy invasion [21]. They are expected to transform health-care and industry [20] By current estimates, this interconnected universe will include 30 billion devices by 2020 [7]. As theses devices become more prevalent, their security is becoming a concern. It is important to investigate and evaluate designs for low-power crypto primitives that have a small memory footprint. As experts in the field put it, the IoTs future will rely on our ability to adequately secure hard-to-secure, resource-sparse devices [3].

ANT+ is a wireless sensor network protocol designed for IoT devices by Dynastream Innovations, Inc which is gaining popularity. ANT+ is marketed as an ultra-low power, efficient and easy to use protocol for sport, home care and medical devices. Hundreds of ANT+ are on the market, from Fitbits to blood glucose monitors, and phones are already being produced with built-in ANT+ capabilities [9].

For our final project, we have analyzed the ANT+ security protocol and searched for a low-energy message authentication code to improve it. Section 2 begins with an overview of the ANT+ Protocol. Section 3 contains a discussion of the security of the ANT+ protocol. We found that ANT+ has made some provisions for encryption, but has no means of cryptographic authentication. This is a concerning vulnerability for devices that are transmitting personal data. In Section 3, we suggest several potential low-energy message authentication codes (MACs) that might be appropriate for augmenting the ANT+ Protocol. In Section 4, we describe the experimental setup we used to evaluate the energy used by these MAC schemes. In Section 5, we describe our methodology and present our results in Section 6.

2 Overview of ANT+ Protocol

ANT is a network protocol designed for Internet of Things (IoT) sensors. Like many other common wireless protocols (802.11, Bluetooth, etc.) it operates in the 2.4 GHz ISM band. Its primary current use is for device-to-device communication between Master devices, typically sensors such as heart rate monitors or geocaching chips, and slaves, such as ANT-enabled watches and cell phones which process sensor data. However, the protocol supports more than a one-to-one master-slave relationship; it supports star, tree, and mesh topologies. These networks are established through a chosen frequency, channel ID, and network key, which combine to create a channel between devices. The process for creating channels is shown in Figure 1.

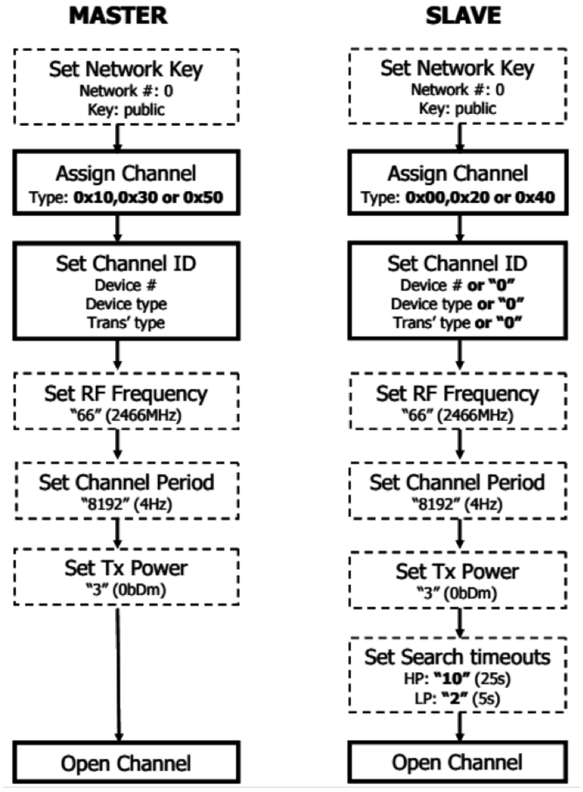
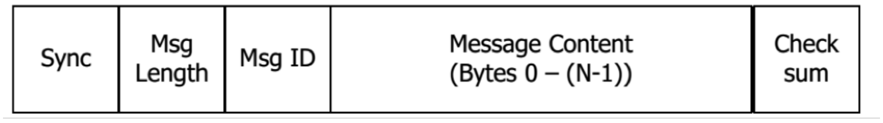


Figure 1: Process to establish a channel between master and slave

ANT typically operates in burst mode, with 64-bit packets of information. Each packet contains header information necessary for message transmission and a check sum to verify message contents. Optionally, ANT can operate in an authenticated mode which allows for the acknowledgement of messages. However, this method simply adds an ACK reply, based on the check sum, from receiving devices; it does not add a cryptographically-secure MACs. Additionally, ANT offers an advanced burst mode of 128-bit packet size that draws more power. The format for these packets is shown in Figure 2. A 64-bit Network Key is required to initiate a channel. This key only secures the creation of the channel; it does not encrypt messages sent within the channel.



Sync	Msg Length	Msg ID	Message Content (Bytes 0 – (N-1))	Check sum

Figure 2: ANT serial message structure

ANT+, by Dynastream Innovations, builds on the ANT protocol by standardizing device profiles, which are set parameters for a list of devices, such as a stride based speed and distance monitor (a pedometer). These profiles assign each type of device to a specific frequency within the ANT band as well setting more technical details, such as the other requirements for initiating a channel shown in the figure above. Furthermore, in order to allow for easy interoperability between the various sensors and possible slave devices (phones, watches, etc.), ANT+ dictates a centrally-managed scheme for Network Keys, allowing devices to easily connect to one another at the cost of the meagr security benefits provided by the Network Key.

3 Security of ANT+

ANT+ has several security vulnerabilities, as discovered by our analysis of the development documentation and source code. Broadly, we categorize these vulnerabilities by the tenets of security they threaten:

3.1 Confidentiality

ANT+ is unencrypted by default. The protocol appears to offer confidentiality through two mechanisms: RF frequency/Channel ID and a network key.

3.1.1 RF frequency/Channel ID

By assigning each communication to a unique frequency and Channel ID, ANT+ appears to the user to offer confidentiality. However, this system only prevents legitimate users from receiving data from unintended IoT devices; it does nothing to prevent malicious interception, as the ANT+ frequencies are assigned by master devices profile (e.g., Stride Based Speed and Distance Monitor).

3.1.2 Network key

Each ANT+ packet is encrypted with a 64-bit Network Key. However, due to the relative short length of this key, and the deterministic nature of the encryption function, this system does not provide adequate security. Moreover, the network key defaults to a public value to allow interaction with other ANT+ devices, so typical development will not provide even this level of security.

However, threats to confidentiality could be prevented through use of ANT+s optional encryption, AES-128 in CTR mode. Unfortunately, there are three usage cases that severely impede the usage of ANT+s AES encryption in the discussed low power applications.

3.1.3 Multi-node networks

ANT+ prefers to use multichannel communication to support multi-node network topologies (the desired IoT end state). However, AES encryption cannot be used in multichannel mode, forcing the usage of single channel communications. While single channel schemes can support multi-node topologies, they become highly power inefficient, as all Master devices (the IoT devices) must operate in continuous scanning mode, which draws significant power and therefore should not be used for devices that have tight power constraints. (ANT+ Message Protocol and Usage, p28)

3.1.4 Low power applications

ANT+ requires the advanced burst method of communication with AES encryption (to support the 128-bit block size), which is more power intensive than the traditional burst communication. Moreover, the AES computation itself is power intensive relative to other algorithms, as discussed further in our results section below.

3.1.5 Low cost or legacy applications

When sourcing ANT+ processors from the available vendors (Texas Instruments, Nordic Semiconductor, and Dynastream Innovations), AES-capable processors typically cost two to three times as much per processor. Thus, in low cost applications, it may not be feasible to implement AES. Moreover, AES capability is a recent development, so older ANT+ processors also lack the capability, forcing implementations that require backwards compatibility to forego AES encryption.

3.2 Availability

The principle threat to ANT+ availability is broad spectrum jamming. Because ANT+ operates on popular frequencies (the 2.4GHz ISM band), technology to affect this communication is widespread. The low power nature of IoT devices worsens this vulnerability, especially given that broadcasting is the most power intensive task for a typical IoT device. However, ANT+ natively supports a frequency agility capability that allows it to reduce/remove interference from fixed-frequency devices (e.g., a wireless access point). This capability is not frequency hopping; it is reactionary only and does not subvert direction finding or transmission capture.

3.3 Integrity

ANT+ provides no cryptographic authentication. Thus, it is possible to forge packets given knowledge of their Network Key, discussed previously, and Device Profile, which is determined by the type of device the attacker would wish to impersonate. If the attacker is impersonating an IoT device that the ANT+ slave has not synced with before, this is trivial, as ANT+ allows for a Trust On First Use (TOFU) system where Channel IDs are stored by the slave device for ease of future sync.

However, this vulnerability should not be associated with the common Man In The Middle (MITM) attack. While it is possible to act as a MITM for a new sync partnership, the nature of point-to-point wireless communication makes it very difficult for an adversary to prevent the two targeted nodes from communicating directly to one another. However, the pseudo-MITM adversary could forge the communication to stop a sync and then initiate new syncs (with forged information) with both targets. This would then have the same effect as a traditional MITM attack; however, it would be noticeable (if not obviously malicious) to the targets. This pseudo-MITM attack is shown in Figure 3.

In general, integrity of the messages are often more important in wireless sensor networks than confidentiality. The confidentiality is important only when we have something to keep secret. However, without integrity, whole a realm of attacks such as cut-and-paste attacks would be possible. Therefore, systems like TinySpec [10] require authentication, while keeping encryption optional. We think that for protocols like ANT+, authentication and integrity of messages should be of utmost importance. Hence, we focus our efforts on purposing low-energy authentication scheme for ANT+.

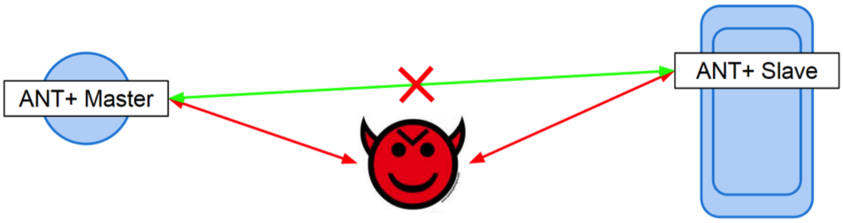


Figure 3: Pseudo-MITM attack scenario in ANT+ protocol

4 Low-Energy MACs

Having established that the ANT+ protocol is lacking cryptographic authentication, we began to search for a low-energy security primitive to mend this security hole. Instead of trying to design a new lightweight message authentication code (MAC) for ANT+, we took advantage of the wealth of research that has already been done on lightweight security. Through a careful literature search, we picked out several low-energy MACs that seemed appropriate for ANT+. We then used a small experiment to decide between the candidate MACs.

4.1 Desireable MAC Properties

A few different design criteria went into our selection of lightweight MACs. Primarily, we were concerned with the energy consumed by the MAC. This has two different components - the energy cost of computation, and the energy cost of transmission. The energy cost of computation refers to the energy used by the CPU as the MAC is calculated. (We were primarily concerned with the energy consumption of the transmitting sensor mode; in our modeled use case the receiving node which gathers sensor data is likely to have a more relaxed energy constraint - receiving nodes are often larger devices with more battery resources, like cell phones, smart watches or laptops.) The energy cost of transmission refers to the energy cost of using the RF transmitter to send the extra bits of MAC, in addition to the data. As can be seen from the figure below, the energy of transmission usually dwarfs the energy of computation. In fact, in most systems sending one bit of data has the same energy cost of 800-1000 cycles of computation [11].

A key design feature, then, was to have a MAC that was as small as possible, while still providing security. We chose to focus on 64-bit MACs for two reasons: A 64-bit MAC provides a good trade-off between size and security (while a few 32-bit and 48-bit MACs do exist, at those sizes MACs provide limited security.) Furthermore, the smallest chunk of data that ANT+ can transmit is 64 bits.

When considering different MAC options, we also paid attention to the energy cost of encryption and ease of use. (ANT+ is designed to be accessible and user-friendly, and we felt it important to maintain this characteristic.) Ultimately, we used experimental data to compare MACs in terms of encryption energy. Other parameters such as memory footprint, or FPGA gate count can be used for comparing MACs, but we felt they were not within the scope of this project.

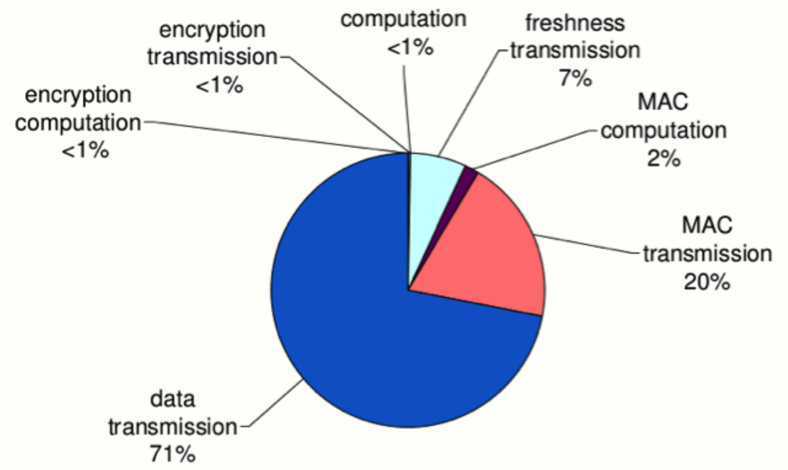


Figure 4: This plot was taken from a realized security protocol for wireless network sensors[5]. It breaks down the energy consumption of an 8-byte CBC-MAC with RC5 for a 30-byte message. A key feature of the plot is the low cost of MAC computation relative to the cost of transmission.

4.2 Considered MAC Schemes

There are three main categories of MACs: those that are block-cipher based (like CBC-MAC), those that are hash function based (like HMAC) and those that are stream-cipher based. Of these three, we focused on block-cipher based MACs. Stream-cipher based MACs generally have a higher overhead associated with initializing the PRG, and we were unable to find a lightweight hash function with a 64-bit digest.

Among block-cipher based MAC schemes, we considered several options, including CMAC [17] (which is essentially CBC-MAC with multiple keys to fix the message-extension attack), PMAC1 (Parallelizable MAC version one) [13] , GMAC (Galois Message Authentication Code) [14] and MARVIN [12]. PMAC1 is a parallelization mode of operation for block-ciphers, GMAC combines the counter mode of operation with Galois multiplication in a structure that follows the Carter-Wegman [15] design, and MARVIN was specifically designed for constrained platforms and follows the ALRED [16] construction. A recent study found GMAC and PMAC to be relatively energy-inefficient [11]. According to the same study, both MARVIN and CMAC compare favorably in terms of energy consumption. Between the two, CMAC is much more familiar to developers (it is commonly used in wireless sensor network applications [cite]). With ease of use in mind, we decided to focus on testing CMAC with different block ciphers.

4.3 Lightweight Block-Ciphers

Figure 1 lists the currently proposed lightweight block ciphers. Of these, many do not have 64-bit blocks. Many others have security concerns. KLEIN can be broken via a truncated differential attack [50]. Piccolo can be broken in a few hours by a biclique meet-in-the-middle attack [51]; HIGHT and TWINE can be broken by the same attack [53]. GOST can also be broken by a meet-in-the-middle attack [52]. XTEA is weak against related-key rectangle attacks [55].

Several other of these block ciphers do not compare favorable in terms

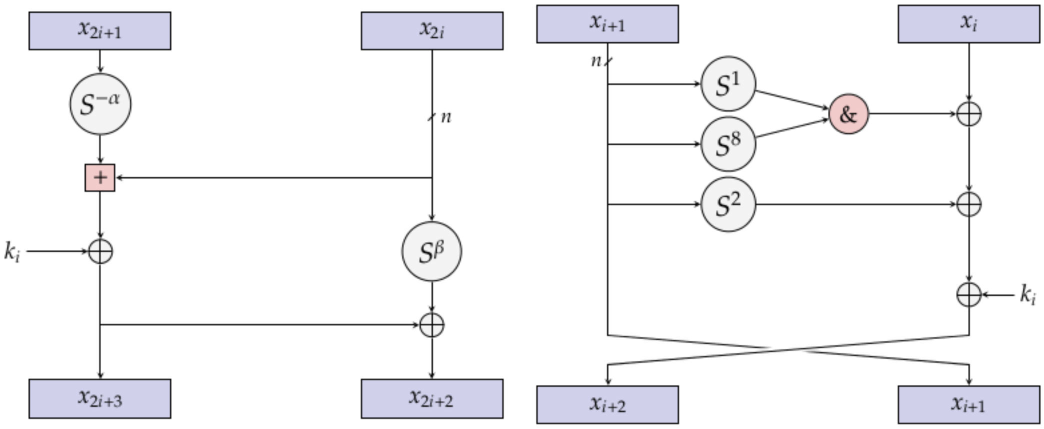


Figure 5: These two diagrams show the round structure of the SPECK (left) and SIMON (right) block ciphers.

of computation time. A recent comparison of several block ciphers found LED, KTAN and KTANTAN to be relatively expensive in terms of computation energy [18].

Of the remaining block ciphers, we selected those which seemed to have the best trade-off between performance and security: SIMON, SPECK, and Skipjack. Since AES is so commonly used, we decided to include it in our experimental tests as well for the sake of comparison, even though it does not have a 64-bit block.

SIMON and SPECK are two sister families of block ciphers developed by the NSA. They are both Feistel Networks with two branches, but they have two different Feistel functions. SIMON was designed to be particularly fast in hardware, and it relies on and, rotation and xor operations. SPECK was designed for software implementations, and it uses a modified Feistel Network where both branches are modified in each round. It uses addition, rotation and xor operations. Diagrams of these block ciphers can be seen in Figure 5.

Skipjack was also developed by the NSA, and later declassified. It is an unbalanced Feistel Network with 32 rounds.

Block Cipher	Block Sizes (bits)	Key Sizes (bits)	Structure	reference
AES	128	128, 192, 256	Substitution-Permutation Network (SPN)	[28]
Chaskey Cipher	128	128	Feistel Network - based	[29]
CLEFIA	128	128, 192, 256	Feistel Network - based	[30]
DESLX	64	128	Feistel Network - based	[31]
GOST revisited	64	256	Feistel Network - based	[32]
HIGHT	64	128	Feistel Network - based	[33]
ITUbee	80	80	Feistel Network - based	[34]
KLEIN	64	64, 80, 96	Stream-cipher-like	[35]
KTAN, KTANTAN	32, 48, 64	254	Stream-cipher-like	[36]
LBlock	64	80	Feistel Network - based	[37]
LED	64	64, 128	SPN	[2]
mCrypton	64	64, 96, 128	SPN	[38]
Noekeon	128	128	SPN	[39]
Piccolo	64	80, 128	Feistel Network - based	[40]
PRESENT	64	80, 128	SPN	[41]
PRIDE	64	128	SPN	[42]
PRINCE	64	128	SPN	[43]
RC5	32, 64, 128	$0 \ldots 2040$	Feistel Network - based	[44]
SEA	96	96	Feistel Network - based	[45]
Skipjack	64 bits	80 bits	Feistel Network - based	[1]
SIMON, SPECK	32, 48, 64, 96, 128	64, 72, 96, 128, 144, 192, 256	Feistel Network - based	[46]
TWINE	64	80, 128	Feistel Network - based	[47]
XTEA	64	128	Feistel Network - based	[48]
Zorro	128	128	Feistel Network - based	[49]

Table 1: This table summarizes the characteristics of the currently proposed lightweight block ciphers

5 Experimental Setup and Methodology

In evaluating the block ciphers we selected based on methodology in Section 4, we focused our efforts on two different aspect of energy consumption. Firstly, we wanted to get estimation for energy it takes the transmitter to transmit one Ant block (64-bit) across a channel. Secondly, we wanted to measure the energy spent by the CPU for computing a MAC. The details of the measurements and estimation is explained in the following subsections.

5.1 Measuring Transmission Energy

In our setup, an android phone broadcasts data via Ant USB stick to other receiver. To measure the transmission energy going across that Ant USB stick, which transmits/receives data, we setup an environment that can intercept the power usage of the transmitter. Namely, we hooked a resistor in between the And USB stick and it's connection to the phone, so that we can measure the voltage across that resistor when we are transmitting data and when we are not transmitting (i.e. idle). Figure 6 shows the setup.

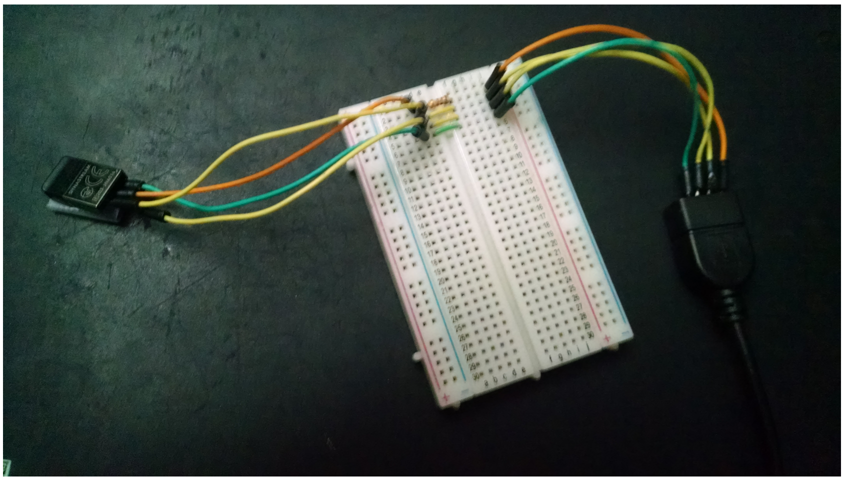


Figure 6: Setup for measuring transmit energy of the transmitter

5.2 Measuring CPU Energy for computation

To measure the energy spent by CPU for computing a MAC using one of the block ciphers, we implemented Block Ciphers into an Android App which broadcast many message along with their MAC to ANT Receiver as shown in Figure 7.

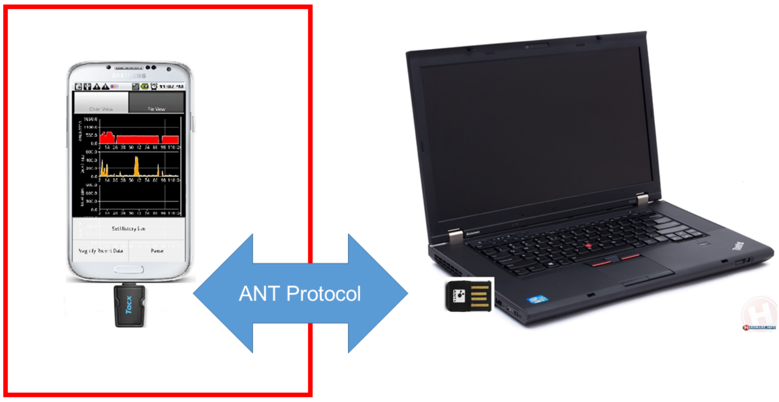


Figure 7: Setup for measuring CPU energy by profiling the phone during computation

We extended the sample Ant application [26] provided by Ant developers to compute CMAC for the message size we select and broadcast large number of messages along with their MAC-s.

As for the implementation of the actual block ciphers, we used the implementations [25] provided by Simplicio, Marcos A., et al. for their survey paper [4]. We copied implementations for AES, Simon, Speck, Skipjack to one C file which we called from the app using Java Native Interface.

When the application starts broadcasting messages, we record the CPU utilization of the app through a profiler called PowerTutor [24]. PowerTutor reads /proc/PID/stat at 1 Hz frequency and calculates CPU utilization based on clock ticks for the PID, and linearly interpolates power from CPU freq parameters specified for the CPU model. It's CPU utilization calculation is similar to tools like

TOP. We modified that PowerTutor application, which could be found at [23], to use the CPU parameters for Galaxy S3 [19][22] and to sample more frequently.

6 Results

6.1 Transmit Energy

To ascertain the energy of transmitting an ANT+ packet, we measured with a multimeter the voltage over a 10 ohm resistor on the ANT+ device's 5.0 V power supply line. When the device was not transmitting, we measured an average voltage drop of 0.1015 V. When the device was transmitting at a rate of 1 Hz, we measured an average voltage drop of .1025 V, so transmission caused an extra 0.001 V drop over the resistor. Usingthe equations $P = V * I = \frac{V_{cc} * V_{drop}}{R}$ and $\bar{E} = \bar{P} \times \Delta t$, we found a rough value for the energy consumptionof transmission to be 500 uJ.

6.2 CPU Energy

After measuring the power usage of the Android app through PowerTutor at 10Hz, we estimated the total energy spent during computation and transmission of **500** Ant messages along with MAC. We use the following equation for estimating total energy.

$$E = \sum_{i=1}^{500} P_i \Delta t \tag{1}$$

To compute the energy per message and its MAC, we simply divide the total energy by **500**. Figure 8 shows the energy spent per message plus its mac computation for different message sizes and block cipher algorithms. For comparison purposes, we also measured the energy when we are not doing any mac computation.

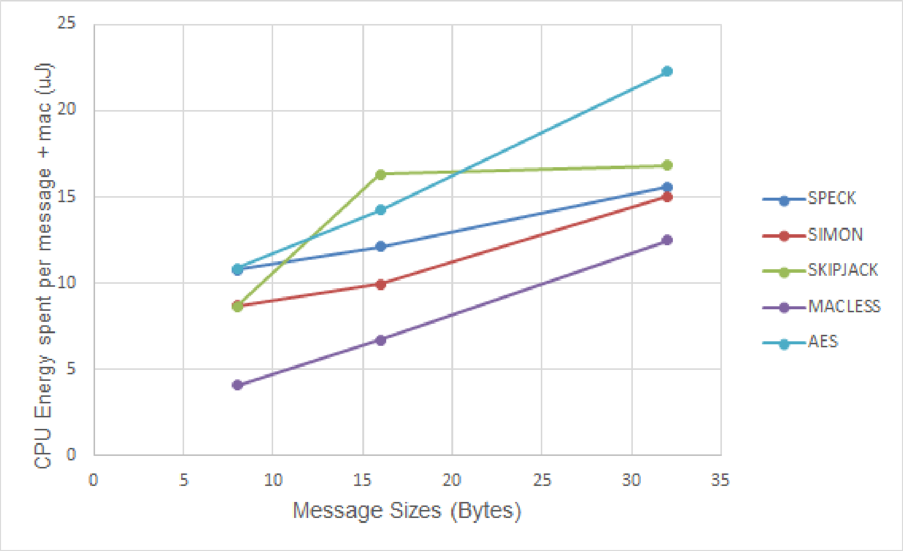


Figure 8: CPU Energy consumption per message and mac computation for different message sizes

To isolate the energy spent per MAC computation, we subtract the energy without MAC from the energy with MAC computation. Using this methodology, we can see in Figure 9 that block cipher Simon seems to be the most efficient in terms of its CPU Energy consumption for computation.

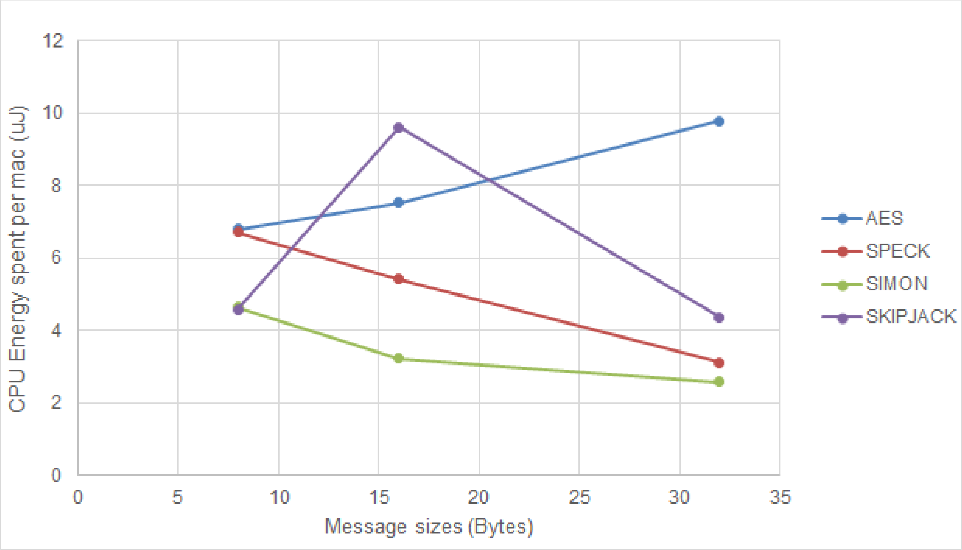


Figure 9: CPU Energy consumption per message and mac computation for different message sizes

Overall, across all the different block cipher implementations we implemented, the energy spent per MAC computation is very small compared to the energy spent on transmission. CPU energy for computation ranged from 2 to 12 uJ whereas the energy spent on transmission was around $500uJ$. This implies that energy spent by CPU for computation of MAC is less than **%3** of the energy for transmission. Therefore, it is justified to say that when considering low-energy MAC-s, one should pick the block size to be as small as possible, and then worry about the computation cost.

7 Conclusion

We were motivated by the emerging Internet of Things (IoT), which is the concept that soon everything will be able to collect data and send it to the Internet through low-energy wireless sensors. As the applications of IoT devices spreads to healthcare and other sensitive fields, is increasingly important that IoT nodes have some level of security in their communication protocol. With this in mind, we analyzed a widely-used protocol for low-power wireless sensor networks called ANT+.

As is discussed in Section 3, ANT+ has several vulnerabilities. In terms of confidentiality, ANT+ is unencrypted by default. It supports AES128 encryption, but only in the particular case of single-channel communication which consumes a lot of energy. More importantly, ANT+ has no cryptographic authentication. For wireless network protocols, providing integrity for the messages is crucial. We reason that the protocol is vulnerable to our pseudo-MITM attacks.

With our security analysis in mind, we focused our efforts on identifying MAC schemes suitable for supplementing the low-energy ANT+ protocol. Since transmission of data is the dominant factor in message energy consumption, a key design criteria was MAC length. Based on our analysis and research of various MAC implementations, we decided that CMAC was the most appropriate MAC algorithm. We identified three light-weight block ciphers that have a small block size and evaluated them in terms of their computation cost. As measured by our setup, the SIMON cipher showed the lowest CPU usage.

We believe wireless sensor network protocols like ANT+ should consider implementing CMAC-based authentication mechanisms into their protocol, since providing integrity is critical to the wide-spread deployment of IoT sensors and its adoption by the public.

References

[1] Skipjack, Nist. n.d. KEA Algorithm Specification, 1998.

[2] Guo, Jian, Thomas Peyrin, Axel Poschmann, and Matt Robshaw. 2011. The LED Block Cipher. In Cryptographic Hardware and Embedded Systems CHES 2011, 32641. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[3] Trappe, Wade, Richard Howard, and Robert S. Moore. *Low-Energy Security: Limits and Opportunities in the Internet of Things.* IEEE Security & Privacy 1 (2015): 14-21.

[4] Simplicio, Marcos A., et al. *Survey and comparison of message authentication solutions on wireless sensor networks.* Ad Hoc Networks 11.3 (2013): 1221-1236.

[5] Perrig, Szewczyk,Tygar, Wen, Culler. *SPINS: security protocols for sensor networks.* Wirel. Netw. 8, 5 (September 2002), 521-534.

[6] Sohraby, K., Minoli, D., Znati, T. *Wireless sensor networks: technology, protocols, and applications*, John Wiley and Sons, 2007 ISBN 978-0-471-74300-2, pp. 203209

[7] Reddy, A. 2014. Reaping the Benefits of the Internet of Things. Teaneck, New Jersey, USA, Cognizant, Cognizant Reports.

[8] Trappe, Wade, Richard Howard, and Robert S. Moore. *Low-Energy Security: Limits and Opportunities in the Internet of Things.* IEEE Security and Privacy 1 (2015): 14-21.

[9] `http://www.thisisant.com/directory/`

[10] Karlof, Chris, Naveen Sastry, and David Wagner. 2004. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems, 16275. SenSys 04. New York, NY, USA: ACM.

[11] Sadaqat Ur Rehman,Muhammad Bilal, Basharat Ahmad, Khawaja Muhammad Yahya, Anees Ullah, Obaid Ur Rehman, *Comparison Based Analysis of Different Cryptographic and Encryption Techniques Using Message Authentication Code (MAC) in Wireless Sensor Networks (WSN*

[12] Simplicio, Marcos A, Pedro Daquino F F S Barbuda, Paulo S L M Barreto, Tereza C M B Carvalho, and Cintia B Margi. 2009. The MARVIN Message Authentication Code and the LETTERSOUP Authenticated Encryption Scheme. Security and Communication Networks 2 (2). John Wiley & Sons, Ltd. 16580.

[13] Khan, N P, and C Boncelet. 2006. PMAC: Energy Efficient Medium Access Control Protocol for Wireless Sensor Networks. In Military Communications Conference, 2006. MILCOM 2006. IEEE, 15.

[14] Dworkin, Morris. 2007. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. US Department of Commerce, National Institute of Standards and Technology.

[15] Wegman, Mark N, and J Lawrence Carter. 1981. New Hash Functions and Their Use in Authentication and Set Equality. Journal of Computer and System Sciences 22 (3): 26579.

[16] Daemen, Joan, and Vincent Rijmen. 2005. A New MAC Construction ALRED and a Specific Instance ALPHA-MAC. In Fast Software Encryption, 117. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[17] Dworkin, Morris. 2005. Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. US Department of Commerce, Technology Administration, National Institute of Standards and Technology.

[18] Cazorla, Mickel, Kevin Marquet, and Marine Minier. n.d. Survey and Benchmark of Lightweight Block Ciphers for Wireless Sensor Networks?

[19] Heilein, John. 2010. Leveraging Advanced Physical IP to Deliver Optimized SoC to Deliver Optimized SoC Implementations at 40nm and below. Artisan Advanced Physical IP by ARM

[20] `http://www.spie.org/x105127.xml`

[21] Bokareva, Tatiana, et al. *Wireless sensor networks for battlefield surveillance.* Proceedings of the land warfare conference. 2006.

[22] `http://www.gsmarena.com/samsung_i9300_galaxy_s_iii-4238.php`

[23] `https://github.com/msg555/PowerTutor`

[24] `http://ziyang.eecs.umich.edu/projects/powertutor/`

[25] `https://github.com/kmarquet/bloc`

[26] `http://www.thisisant.com/developer/ant/ant-in-android`

[27] `htt://www.nuee.nagoyau.ac.jp/labs/tiwata/omac/omac.html`

[28] Daemen, Joan, and Vincent Rijmen. 1998. AES Proposal: Rijndael. `http://158.38.101.8/DisMath/rijnddoc.pdf`

[29] Mouha, Nicky, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. 2014. Chaskey: An Efficient MAC Algorithm for 32-Bit Microcontrollers. In Selected Areas in Cryptography – SAC 2014, 30623. Lecture Notes in Computer Science. Springer International Publishing.

[30] Shirai, Taizo, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. 2007. The 128-Bit Blockcipher CLEFIA. In Fast Software Encryption, 18195.

[31] Leander, Gregor, Christof Paar, Axel Poschmann, and Kai Schramm. 2007. New Lightweight DES Variants. In Fast Software Encryption, 196210. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[32] Poschmann, Axel, San Ling, and Huaxiong Wang. 2010. 256 Bit Standardized Crypto for 650 GE–GOST Revisited. In Cryptographic Hardware and Embedded Systems, CHES 2010, 21933. Springer.

[33] Hong, Deukjo, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bon-Seok Koo, Changhoon Lee, et al. 2006. HIGHT: A New Block Cipher Suitable for Low-Resource Device. In Cryptographic Hardware and Embedded Systems - CHES 2006, 4659. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[34] Karako, Ferhat, Hseyin Demirci, and A Emre Harmanc. 2013. ITUbee: A Software Oriented Lightweight Block Cipher. In Lightweight Cryptography for Security and Privacy, 1627. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[35] Gong, Zheng, Svetla Nikova, and Yee Wei Law. 2012. KLEIN: A New Family of Lightweight Block Ciphers. In RFID. Security and Privacy, 118. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[36] De Canniere, Christophe, Orr Dunkelman, and Miroslav Kneevi. 2009. KATAN and KTANTANa Family of Small and Efficient Hardware-Oriented Block Ciphers. In Cryptographic Hardware and Embedded Systems-CHES 2009, 27288. Springer.

[37] Wu, Wenling, and Lei Zhang. 2011. LBlock: A Lightweight Block Cipher. In Applied Cryptography and Network Security, 32744. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[38] Lim, Chae Hoon, and Tymur Korkishko. 2006. mCrypton–a Lightweight Block Cipher for Security of Low-Cost RFID Tags and Sensors. In Information Security Applications, 24358. Springer.

[39] Daemen, Joan, Michal Peeters, Gilles Van Assche, and Vincent Rijmen. 2000. Nessie Proposal: NOEKEON. In First Open NESSIE Workshop. http://cryptospecs.googlecode.com/svn/trunk/symmetrical/specs/noekeon.pdf.

[40] Shibutani, Kyoji, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. 2011. Piccolo: An Ultra-Lightweight Blockcipher. In Cryptographic Hardware and Embedded Systems CHES 2011, 34257. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[41] Bogdanov, A, L R Knudsen, G Leander, C Paar, A Poschmann, M J B Robshaw, Y Seurin, and C Vikkelsoe. 2007. PRESENT: An Ultra-Lightweight Block Cipher. In Cryptographic Hardware and Embedded Systems - CHES 2007, 45066. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[42] Albrecht, Martin R, Benedikt Driessen, Elif Bilge Kavun, Gregor Leander, Christof Paar, and Tolga Yaln. 2014. Block Ciphers–Focus on the Linear Layer (feat. PRIDE). In Advances in Cryptology–CRYPTO 2014, 5776. Springer.

[43] Borghoff, Julia, Anne Canteaut, Tim Gneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R Knudsen, Gregor Leander, et al. 2012. PRINCE–a Low-Latency Block Cipher for Pervasive Computing Applications. In Advances in Cryptology–ASIACRYPT 2012, 20825. Springer.

[44] Rivest, Ronald L. 1995. The RC5 Encryption Algorithm. In Fast Software Encryption, 8696. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[45] Standaert, Franois-Xavier, Gilles Piret, Neil Gershenfeld, and Jean-Jacques Quisquater. 2006. SEA: A Scalable Encryption Algorithm for Small Embedded Applications. In Smart Card Research and Advanced Applications, 22236. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[46] Beaulieu, Ray, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. 2013. The SIMON and SPECK Families of Lightweight Block Ciphers. IACR Cryptology ePrint Archive 2013: 404.

[47] Suzaki, Tomoyasu, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. 2011. Twine: A Lightweight, Versatile Block Cipher. In ECRYPT Workshop on Lightweight Cryptography, 14669.

[48] Needham, Roger M, and David J Wheeler. 1997. Tea Extensions. Report, Cambridge University, Cambridge, UK (October 1997). http://www.club.cc.cmu.edu/~ajo/docs/xtea.pdf.

[49] Grard, B, Vincent Grosso, M Naya-Plasencia, and Franois-Xavier Standaert. 2013. Block Ciphers That Are Easier to Mask: How Far Can We Go? In Cryptographic Hardware and Embedded Systems - CHES 2013, 38399. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[50] Lallemand, Virginie, and Mara Naya-Plasencia. 2014. Cryptanalysis of KLEIN. In Fast Software Encryption-FSE 2014. http://fse2014.isg.rhul.ac.uk/slides/slides-07_5.pdf.

[51] Wang, Yanfeng, Wenling Wu, and Xiaoli Yu. 2012. Biclique Cryptanalysis of Reduced-Round Piccolo Block Cipher. In Information Security Practice and Experience, 33752. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[52] Dinur, Itai, Orr Dunkelman, and Adi Shamir. 2012. Improved Attacks on Full GOST. In Fast Software Encryption, 928. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[53] Hong, Deukjo, Bonwook Koo, and Daesung Kwon. 2012. Biclique Attack on the Full HIGHT. In Information Security and Cryptology - ICISC 2011, 36574. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[54] oban, Mustafa, Ferhat Karako, and zkan Bozta. 2012. Biclique Cryptanalysis of TWINE. In Cryptology and Network Security, 4355. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[55] Lu, Jiqiang. 2008. Related-Key Rectangle Attack on 36 Rounds of the XTEA Block Cipher. International Journal of Information Security 8 (1). Springer-Verlag: 111.