A Comparison of Two Modes for AEAD Services in Wireless Sensor Networks

Albin Eldstål-Damlin, Laurynas Riliskis
Department of Computer Science, Electrical and Space Engineering
Luleå University of Technology
971 87 Luleå, Sweden

***Abstract*—In the context of resource-limited sensor motes, security is often discarded or compromised upon. This report presents tests on SCM and CCM, two modes of operation for ensuring *confidentiality* and *authenticity* of messages passed between nodes in a wireless sensor network. The modes are compared with regards to time complexity and energy efficiency when run on the iRoad Mulle. We show that SCM is both faster and more energy efficient than CCM, as well as more suited for multi-hop routing topologies.**

I Introduction

Often, the security aspects if a research system are neglected for the sake of simplicity; a proof-of-concept model does not face the same requirements as a real life implementation. Nodes in Wireless Sensor Networks are built with energy-efficiency as their primary goal, therefore computational resources are heavily constrained. Such platforms require new, novel approaches when implementing security measures on the nodes. In this document we compare SCM and CCM, two symmetric encryption modes built upon the Advanced Encryption Standard (AES). Both are considerably cheaper than available PK algorithms and provide full AEAD (Authenticated Encryption with Associated Data) services. We present results showing the energy consumption of each method and their impact on the throughput of a wireless sensor network.

Contents

II. The Modes

A. AEAD Services

When dealing with computer security in general and cryptography in particular, it is important to remember that there are several separate aspects commonly collected under the *security* umbrella. *Confidentiality* is the guarantee that only the intended recipient of a message may see its contents. *Integrity* is the guarantee that the received message is the same as that intended by the sender. *Authenticity* is the guarantee that a message is sent from the indicated source.

The concept of Authenticated Encryption with Associated Data (AEAD) is to ensure *Confidentiality* by encrypting each message. The *Integrity* of each message is ensured by calculating a Message Authentication Code (MAC), using same secret shared key as the encryption mechanism. Furthermore, the MAC scheme employed

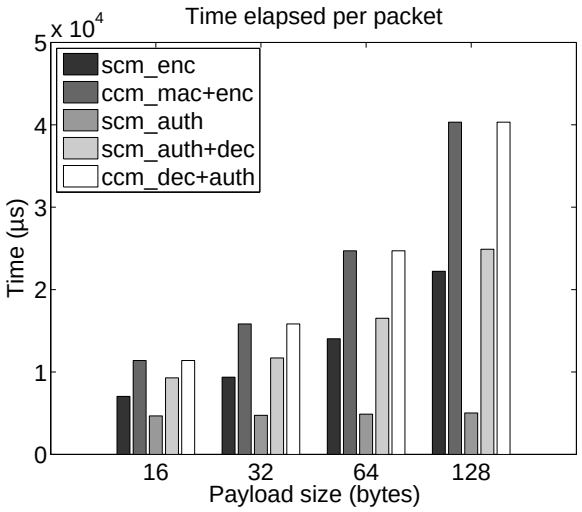

Fig. 1: Time consumption of operations

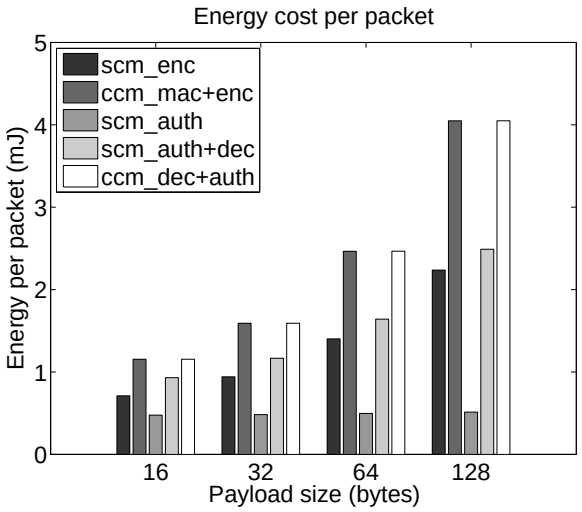

Fig. 2: Energy consumption of operations

includes some non-confidential *Associated Data* (source, destination, sequence numbers, etc.) in the calculation which help satisfy the *Authenticity* requirement.

If desired, some or all of the *Associated Data* may be implicitly inferred nonce data (i.e. unique data not transmitted as part of the message but still known to all authorized parties), part of the metadata from lower communications layers, etc. One good use for the *Associated Data* field is to authenticate the source and destination nodes of a message, which must remain unencrypted for routing purposes. The same data must be available to any intermediate nodes checking the authenticity of a message.

In this document we compare two modes for AEAD, SCM [2] and CCM [2]. Both are based on multiple application of AES and differ mainly in their approach to message authentication.

Both modes employ symmetric encryption based on pre-shared keys.

B. SCM

SCM employs the Encrypt-then-Mac method of operation. In such a scheme the MAC is calculated as a function of the encrypted payload. SCM Calculates the MAC as part of the encryption operation. In addition to the plaintext payload which is to be encrypted, SCM accepts a data *header* which is part of the authenticated data (i.e. verifiably correct) but not encrypted.

Encryption with SCM is symmetric; re-encrypting an encrypted message with the same key yields the original plaintext. The MAC field calculated when performing such a decryption has no meaning.

When encrypting, authenticating or decrypting, two encryption keys and one *tweak* are used, all 16 bytes of length. With the two keys and the tweak pre-shared between a transmitting and a receiving node, the transmitted message contains three parts: any used header data, the ciphertext and a four byte MAC.

C. CCM

CCM employs the Mac-then-Encrypt method of operation, meaning the authentication code for the plaintext is calculated before encryption takes place. Similarly to SCM, CCM allows authenticated but non-confidential *header* data in addition to the encrypted message payload. In contrast to SCM, it uses a single 16-byte key for all operations.

CCM splits the preparation of a new message into two operations; MAC and Encrypt. As with SCM the encrypt operation is symmetric. Calculating the MAC for this plaintext and comparing it to the one received with the ciphertext proves or disproves the integrity and authenticity of the message.

A message protected with CCM contains three parts: any necessary *Associated Data*, the ciphertext and the MAC

	SCM (μs)	CCM (μs)
Encrypt+MAC	14039	24712
Verify	4886	24712
Verify+Decrypt	16525	24712

TABLE I: Time consumption of typical operations with 64 byte payload

III. Experiment setup

All tests were carried out on an iRoad Mulle [1] sensor mote, running a development port of TinyOS 2 [3] modified to allow 80 byte packet payloads instead of the standard 28. For the tests, both CCM and SCM were ported from the reference C implementation to nesC [4] in order to run under TinyOS in the accustomed split-phase fashion. A special *decrypt* operation was implemented for SCM, which verifies the MAC and decrypts a message in a single operation. This is slightly faster than the default authenticate+encrypt, which would unnecessarily perform an extra MAC computation in the encrypt step. No such adaptation was necessary for CCM, whose MAC and Encrypt operations are fully disjoint.

All tests were peformed with the RF212 radio transceiver set to the default 40 kbit/s transmission rate.

A. Energy Consumption

The mote to be measured was connected to a current measurement board, connected to a PC. The measurement board acted as the power supply for the mote while monitoring the energy consumption with a set sampling rate of 204.8 kHz. The measurement board used an operation amplifier to measure 100 times the voltage drop over a 1 ohm resistor, which allowed the operating current to be calculated. The PC was used to log the recorded data. We tested the following six operations:

- SCM encryption and MAC calculation of a payload
- SCM MAC verification of the encrypted data
- SCM decryption and verification of the same payload
- CCM MAC calculation for a payload
- CCM encryption of the same payload
- CCM decryption of the same payload

Each performed test involved running six operations with a 1 second delay between initiation; the delay was inserted to clearly delimit the operations when monitoring the power consumption.

For both methods, the same initial payload value of increasing byte values was used in order to be able to verify the correctness of the finally decrypted data. The mote itself was programmed to measure the time consumed by each performed operation with microsecond precision and store the recorded time in memory.

After testing the six operations, another 1 second pause was induced before validating the decrypted payloads and reporting a summary of the mote's measured values on the serial interface.

When comparing SCM to CCM, it is important to consider that the SCM encrypt operation also calculates the MAC, while CCM splits this into two separate operations. Therefore an additional value, CCM mac+enc, is presented to aid in comparison of the two. Furthermore, to validate the MAC, a CCM system must fully decrypt the message while an SCM system may validate the encrypted payload directly. For a multi-hop network, this has serious implications; the relay nodes have to perform a more expensive set of operations with CCM in comparison to SCM in order to determine the validity of a message before allowing it to propagate.

For both methods, the two operations (decrypt and authenticate) performed by a recipient to decrypt and verify the received message are presented as one combined value to match realistic use.

B. Throughput

Two separate notions of throughput were evaluated for each method; multi-hop and star topology.

In the multi-hop tests (topology shown in Fig. 3) a chain of four relay nodes were situated between the source and destination nodes. The source node would perform encryption and MAC calculation on a message, send it through the chain and listen for traffic. When the last relay was overheard transmitting the message to the final destination, the next message was sent from the source node. This was done to eliminate radio collisions by ensuring that only one message was ever in transit at any given time.

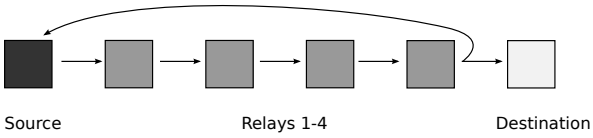

Fig. 3: The Chain topology used for multi-hop throughput tests

Each intermediate node performed verification on messages before passing them on. The destination node decrypted and verified the message, both by MAC and by comparing the plaintext to the known initial value. I the case of SCM, the message was only decrypted if the authentication succeeded. 20 seconds after receiving the first message, the destination node would stop listening for more and report the total number of received valid and invalid packets over the serial link to a connected PC. Each test was repeated four times.

For the star topology tests (topology shown in Fig. 4), five source nodes encrypted and transmitted messages to a single sink. As soon as a source node had fully transmitted a message, the next was prepared and transmitted immediately. If a new message arrived at the sink before a previously received one had been fully processed, the new message would be dropped and counted as such. After 20 seconds the destination would report the total number of valid and invalid messages received from each source node as well as the total number of messages dropped due to processing delay.

IV. Experiment results

A. Time Consumption

Clear differences can be seen between the two methods in terms of processing time. One important difference between SCM and CCM is that the authenticate operation of SCM is nearly constant in time consumption regardless of the payload size. As a result of this the difference between CCM authentication and SCM authentication increases with the size of the payload, as does the difference between full decryption+authentication times. It is also clear from the data that the penalty of decrypting the entire message before authenticating with CCM is quite heavy when compared to SCM.

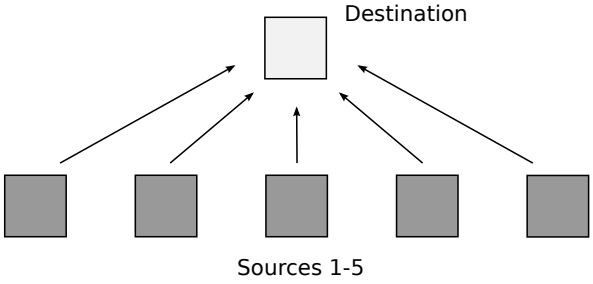

Fig. 4: The Star topology used for multi-hop throughput tests

Table I shows that the time consumption for all three basic message operations is the same when employing CCM, since the operations performed are the same; they are only performed in a different order.

B. Energy Consumption

As both encryption schemes essentially perform a series of AES operations, their levels of hardware utilization while working are very similar. Thus that a faster operation is also more energy efficient, which is confirmed by the time/energy consumption data shown in Fig. 1 and Fig. 2. Further proof of this is visible in figures 5 and 6 which show the current signatures of MAC calculation for a single 32 byte packet using SCM and CCM, respectively. The peak current draw is very similar between the two, while the time consumption is a significantly differentiating factor.

C. Throughput

As indicated by the time consumption data presented, SCM is more efficient than CCM in all the tested scenarios. Depending on the payload size, encryption+authentication using SCM has a time consumption roughly 60% of that shown for CCM. This assertion is further strengthened by the throughput data shown in Fig. 7 and 8. The difference in throughput observed in the chain test can be mostly attributed to the large extra cost of decrypting the message in order for the intermediate nodes to verify the MAC for CCM. In the case of the star topology test, the large number of dropped messages indicates that the cause is the more expensive decryption and verification on the receiving end. Further indication of this is that the total number of transmitted messages is not significantly different between CCM and SCM.

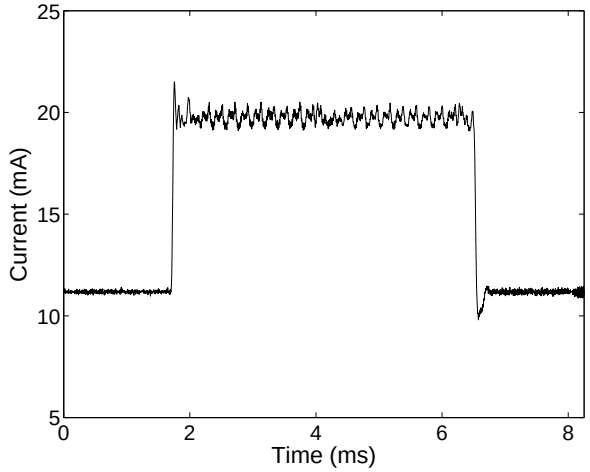

Fig. 5: Current draw for SCM authenticate operation

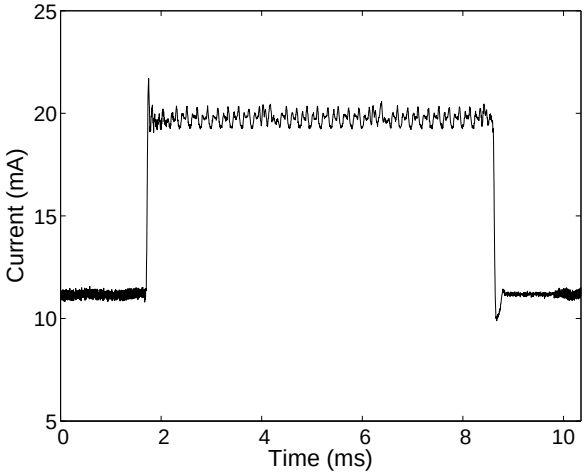

Fig. 6: Current draw for CCM MAC operation

It is noteworthy that SCM achieved throughputs very close to those of the plaintext control test. This does not, however, mean that SCM is as fast as unencrypted communications; it means that both plaintext and SCM are fast enough that the 40 kbit/s radio transmission rate and the TinyOS ActiveMessage network stack become the bottleneck.

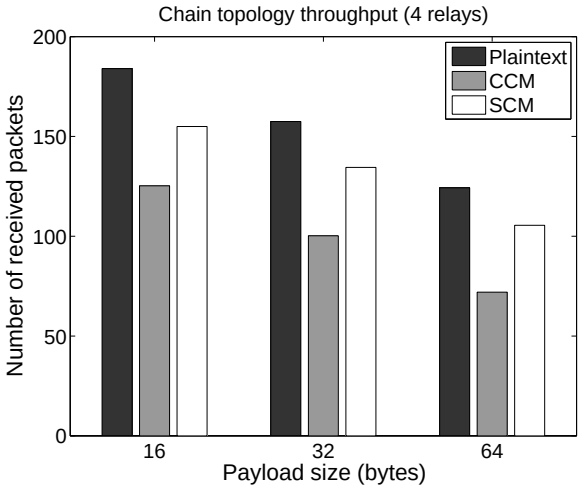

Fig. 7: Packet throughput in a multi-hop topology

V. Conclusion

It is clear from the tests that SCM is superior in running time, therefore also more energy efficient. Depending on the payload size chosen, SCM has a power consumption of 30-60 percent of CCM. While a more exhaustive throughput test at higher transmission rate would further clarify the exact performance, the advantage of the Encrypt-then-Mac approach is clear. The implementation of both modes is similar in complexity and both rely on an existing AES implementation which may be accelerated by special-purpose hardware. Both modes rely on pre-shared keys for their security, which is usually acceptable in the context of WSN since all participating nodes are typically under ”friendly” control at a centralized location during programming/preparation for deployment.

References

[1] The mulle - a wireless sensor node. http://www.eistec.se/mulle.php.

[2] A. Adekunle and S. Woodhead. An efficient authenticated-encryption with associated-data block cipher mode for wireless sensor networks. In Evgeny Osipov, Andreas Kassler, Thomas Bohnert, and Xavier Masip-Bruin, editors, *Wired/Wireless Internet Communications*, volume 6074 of *Lecture Notes in Computer Science*, pages 375–385. Springer Berlin / Heidelberg, 2010.

[3] TinyOS Community. Tinyos community forum. http://www.tinyos.net/, November 2009.

[4] David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, and David Culler. The nesc language: A holistic approach to networked embedded systems. In *Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation*, pages 1–11. ACM Press, 2003.

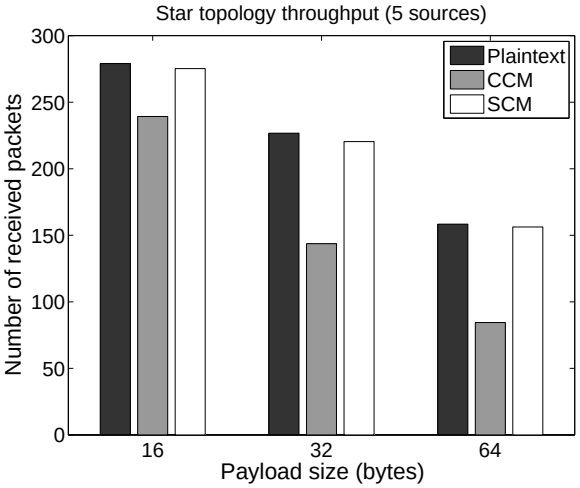

Fig. 8: Packet throughput in a star topology

	Received	**Dropped**
Plain run 1	160	0
Plain run 2	159	0
Plain run 3	160	0
Plain run 4	155	0
CCM run 1	84	72
CCM run 2	84	72
CCM run 3	84	70
CCM run 4	85	71
SCM run 1	157	0
SCM run 2	157	0
SCM run 3	155	0
SCM run 4	156	0

TABLE II: Mean packet counts per source for 64 byte payloads in star topology test