aheM: Additively Homomorphic Encryption for the Moo

Michael Rushanan[1], Denis Foo Kune[2], and Kevin Fu[2]

[1]Johns Hopkins University, `micharu1@cs.jhu.edu`
[2]University of Michigan, `dfkune@umich.edu,kevinfu@umich.edu`

Homomorphic encryption is malleable by design. It enables certain arithmetic operations to be performed on encrypted data by an untrusted party. With respect to passively powered computational RFID tags, such as the Moo, homomorphic encryption is especially useful as tags allowably communicate within a chain of untrusted, and possible volatile, peers. Thus, we propose aheM, an Additively Homomorphic Encryption Scheme for the Moo that extends the work of [1] to the CRFID space.

The initial parameter of aheM is that of a trust assumption; we must assume that fresh (i.e., new or physically accessible) Moos and firmware flashing tools (FFT) are trusted. The FFT maintains the master secret used to key a SHA256-HMAC[1] key derivation function (KDF). That KDF takes a unique statically defined serial number sn_i and outputs the secret key of some Moo$_i$, where sk_i = SHA256-HMAC(sn_i). The sk_i and sn_i are then flashed to the firmware of Moo$_i$.

When the Moo performs its first round of computation, it will generate a random 128bit IV and create a key stream using the MSP430 optimized AES-CTR mode block cipher, k'_i = TI-AES-CTR$_{sk_i}$(IV)[2]. The increment operation on input IV will be denoted as ctr$_i$. Next, the Moo will use k'_i to encrypt its sensor data with the additively homomorphic encryption function [1] as follows, $C_j = Enc_{k'_i}(m_j, M) = m_j + k'_i (\text{mod } M)$. Lastly, the Moo will pack a 96bit URI, called the electronic product code (EPC), with the following values: (SensorID[1byte], C_j[6byte], SensorCtr[2bytes], ctr$_i$[1byte], sn_i[2bytes]).

The EPC is sent to the RFID reader whom forwards the EPC to the cloud. The cloud first indexes unencrypted metadata (e.g., SensorID) to provide efficient sort to the actuator. Next, the cloud computes the aggregation function on all ciphertexts of all Moos as, $C_a = \sum_{i=0}^{n} \sum_{j=0}^{m} C_{i,j}$. This is a significant result of aheM as all tag data has been securely transmitted to the cloud and arithmetically operated on without requiring a notion of trust (i.e., no decryption key was provided).

As the trusted endpoint, the actuator connects to the cloud and requests both the aggregated data and associated metadata (i.e., the unique sn_i and a list of ctr$_i$ per Moo$_i$). Using the master secret, the actuator computes sk_i and then a list of k'_i per Moo$_i$. Each k'_i list can be summed to K and then the aggregate key is computed as the summation of all K, $K_a = \sum_{L=0}^{n} K_L$. With K$_a$, the actuator can finally decrypt the aggregated results, $\text{Dec}_{Ka}(C_a, M) = C_a - K_a (\text{mod } M)$.

References

[1] Claude Castelluccia et al. Efficient Aggregation of encrypted data in Wireless Sensor Networks. 2005.

[1]We only require 128bits due to MSP430 limitations.
[2]Note that we can generate a very long keystream with the linearly incremented counter starting at IV.