Lightweight Stream Encryption for the Internet of Things

Security is one of THE hot topics today in the Internet of Things (IoT). The IoT itself is still in its infancy and the lack of security has quickly become one of its major growing pains. There have been well publicized security breaches of consumer devices that include video from wireless baby monitors being hijacked and posted on the Internet and home automation systems that reveal whether a home is occupied or not. There have been incidents where systems were breached to demonstrate their vulnerabilities and potentially deadly outcomes. In one case, "hackers" were able to take control of a Jeep Cherokee through the WiFi network in its on-board infotainment systems and apply the brakes, kill the engine, cause the vehicle to ignoring steering wheel input, etc. Less well publicized are security breaches with industrial equipment, some of which have made entire industrial facilities non-operational. These types of security breaches are rarely made public for obvious reasons.

Over 40 billion "Internet of Things" (IoT) devices are expected to be deployed globally by the year 2020. By definition these devices are directly or indirectly connected to the Internet. The majority of these devices are network connected via a wireless radio such as Bluetooth or WiFi and access the Internet through a smart phone or some type of "gateway". Most wireless radios feature some form of optional encryption but in order to provide real security encryption needs to be end-to-end (from device to server and vice versa). The built-in encryption in a Bluetooth radio for instance only protects that data on the Bluetooth link. If the data is then sent over the Internet or other network it is unprotected at that point.

Secure data transmission is important if not critical for most IoT devices. However, there are a number of issues that must be overcome in providing data security for IoT devices. The key issues of concern here are:

- A large percentage of these devices are expected to be powered by batteries or small energy harvesting power sources so low power operation is crucial. The majority of these devices are based on small microcontrollers with limited code and data memory spaces with little ability to expand these memory sizes without changing to more expensive parts that consume more power.
- Traditional encryption techniques were developed to run on at least a personal computer so memory requirements and execution time were not important considerations. This and other aspects of traditional encryption techniques make them impractical if not impossible to use on the small microcontrollers typically used for IoT devices.
- Many of the companies producing IoT devices and systems have little or no experience with security in general and specifically data encryption. Since traditional encryption techniques aren't suitable for these devices, companies are faced with providing no encryption or developing their own. Good encryption algorithms are expensive to develop and their effectiveness is often directly proportional to the time spent researching and testing.

There is little that can be done about the low power nature and the small microcontrollers used in small IoT devices. These are the technologies that have made such products possible. They must be acknowledged as a reality for IoT devices and accepted as the conditions and parameters that data security must be delivered within.

As stated earlier, traditional encryption techniques have a number of drawbacks for small IoT devices:

- Traditional encryption algorithms were intended primarily for document encryption, to be performed on at least personal computers so neither memory size nor execution time were major considerations in their development (if considered at all).

- Traditional encryption algorithms were intended to encrypt extremely large amounts of data compared to the short messages used in small IoT applications (megabytes versus several bytes). Since eliminating repetition in encryption ciphers is critical to preventing an algorithm from being hacked or the security keys from being learned, very long keys, complex algorithms and multiple rounds of encryption are used to prevent repetition over megabytes of data.
- Newer algorithms tend to be "block" oriented and are applied to fixed sized blocks of data to facilitate fast matrix arithmetic operations. Block encryption also allows an algorithm to take advantage of multi-core processors and multiple processor systems which is of no concern for a small IoT device. The blocks sizes are often larger than the amounts of data a small IoT device would generally transmit at one time so extra time would be spent encrypting data that wouldn't be transmitted. A "stream" or byte oriented encryption algorithm is much more suitable for IoT devices.
- Most algorithms require a memory footprint that is relatively large for most micros used in small IoT applications. AES (Advance Encryption Standard) is a common encryption algorithm and is usually implemented using 4Kbytes of code space just for lookup tables of cipher data. This is more code space than some small microcontrollers used for simple IoT applications would have. For AES this space can be reduced to 1Kbytes at the expense of additional processing time (which is already too long).
- Most algorithms are expensive in terms of processing times. For battery powered IoT devices, maximizing battery life is critical and the time required to encrypt data with a traditional encryption algorithm can easily exceed the time required for a device to perform its primary function. For AES encryption, each 32 byte block of data typically uses 8 rounds of encryption and each round requires 16 table lookups and 12 32-bit logical operations, followed by four more 32-bit logical operations. On an 8-bit micro this would result in over 1,500 memory accesses (each taking from 1 to 8 clock cycles to execute depending on the micro) plus the time required for the code to execute the algorithm. For a micro running at 4Mhz and 2 clocks per memory access, this equates to around 25 microseconds per byte of data encrypted.

It should be noted that AES is considered a small footprint, fast encryption algorithm so other traditional algorithms would be even worse than the memory requirements and performance numbers above. To be practical for a typical small IoT device, the code space for the encryption algorithm should require well under 1Kbyte of code space and take no more than a few microseconds per byte of data.

Since traditional encryption algorithms aren't suitable for most IoT devices, developers have the choice of developing an encryption algorithm in-house or not to provide any data security. For the reasons discussed earlier, traditional encryption algorithms don't even provide a good starting point for developing an encryption algorithm and very few IoT developers have any type of data security expertise. Taking into account the research and extensive testing that is required in addition to the code development, the cost to develop an encryption algorithm can easily run over well over $30K in fully-burdened engineering time. When completed, the algorithm may present security risks and/or suffer the same issues that traditional encryption algorithms have for IoT devices. Cutting corners on the research and testing almost always results in security vulnerabilities.

The result is most companies choose not to provide any data security in their products or simply rely on the encryption provided by a wireless radio module. This may appease some unknowledgeable customers but end-to-end encryption between the device and server is required to provide true data security.

LSE Technologies has developed the family of LSET (Lightweight Stream Encryption Technology) encryption algorithms specifically for use in IoT applications. We offer several C source code packages to choose from based on your application's security needs and processing resources. These algorithms are based on proven encryption

techniques and were developed specifically to operate with small memory footprints (code and data) and execute efficiently on small microcontrollers while providing adequate security for the short messages common in IoT applications. We also provide functions for random number generation since it is a major element in secure encryption yet it can be very difficult to achieve a high degree of randomness in software, particularly in an efficient manner on a small micro.

The primary benefits to using the LSET encryption algorithms include:

- Algorithms are optimized for small memory sizes and fast execution times.
- Algorithms are based on symmetric keys constructed using customer provided data. Even if the algorithm is hacked someone would need to learn the customer specific data in order to compromise security.
- Several versions are available, tailored to different size micros and the level of security needed for a specific application.
- The algorithm source code is easily integrated into new or existing software (given adequate code and data memory space).
- Much lower cost than developing an encryption algorithm in-house.