Comparative Analysis of Cryptography Library in IoT

Uday Kumar  
Tech Mahindra Limited  
Chennai, India  
udaykumar@techmahindra.com

*Tuhin Borgohain  
Department of Instrumentation Engineering, Assam Engineering College  
Guwahati, India  
borgohain.tuhin@gmail.com

Sugata Sanyal  
Corporate Technology Office, Tata Consultancy Services  
Mumbai, India  
sugata.sanyal@tcs.com

Abstract  
The paper aims to do a survey along with a comparative analysis of the various cryptography libraries that are applicable in the field of Internet of Things (IoT). The first half of the paper briefly introduces the various cryptography libraries available in the field of cryptography along with a list of all the algorithms contained within the libraries. The second half of the paper deals with cryptography libraries specifically aimed for application in the field of Internet of Things. The various libraries and their performance analysis listed down in this paper are consolidated from various sources with the aim of providing a single comprehensive repository for reference to the various cryptography libraries and the comparative analysis of their features in IoT.

Keywords  
ECC, wolfSSL, RELIC, AvrCryptoLib, TinyECC, WiseLib

1. INTRODUCTION  
The implementation of encryption and decryption in the field of cryptography provides a solid means of relaying messages to and fro between users without the added risk of the message being compromised to unwanted personnel. Such encryption-decryption operations are performed by various ways ([3], [7], [15]) through the use of specific set of algorithms. A cryptography library is a sort of repository of the various algorithms available for cryptographic purposes, which provides the added function of categorising the multitudes of algorithms into specific collections based on their performance capacities and functions.

In the field of IoT, microprocessors and embedded devices with low computational power play the vital role of exchange of information using the internet infrastructure. Such constraints to computational capabilities and the necessity of secure exchange of information calls upon the need to implement algorithms specifically optimized to run in resource constrained environments. As such cryptography libraries aimed for use in microprocessors and embedded devices play a very important role for providing the necessary security layers to IoT devices and securing up the overall IoT infrastructure.

2. OVERVIEW:  
In this paper Section 3 will briefly introduce the various cryptography libraries available for encryption in general. It will also list all the encryption algorithms available in the various cryptography libraries. In section 4, we will discuss in details the various cryptography libraries in IoT. In section 5, we will do a comparative analysis amongst the various cryptography libraries discussed in section 4 based on their unique features. We conclude the paper in section 6.

3. CRYPTOGRAPHY LIBRARY  
There exist numerous cryptography library encompassing multitudes of encryption algorithms which can be implemented for encryption of different messages in various fields. These cryptography libraries enable the implementation of various security measures ([11]) through the use of the containing algorithms. Some of the most prominent cryptography library ([5]) along with their encryption algorithms is listed below:

i. Borzoi: The “borZoi” cryptography library implements an algorithm based on elliptic curves (as such known as Elliptic Curve Cryptography Library) ([4], [9], [10], [14], [36]). It implements the following algorithms which ranges over a finite field bearing a characteristic 2 (GF2m) ([1]):
   a. ECDSA (Elliptic Curve Digital Signature Algorithm)
   b. Elliptic Curve Diffie-Hellman Key Agreement Scheme
   c. ECIES (Elliptic Curve Integrated Encryption Scheme)

borZoi is also implemented with AES Symmetric encryption scheme and one other algorithm to produce SHA-1, its digital signature which are as follows ([11]):
   a. AES (Rijndael) Symmetric Encryption Scheme
   b. SHA-1 hash algorithm

ii. Crypto++ : Written in C++, this cryptography library implements various algorithms ranging from authenticated encryption schemes (like GCM, CCM etc.) to algorithms based on elliptic curves (like ECDSA, ECNR etc) ([13]). The various algorithms implemented by Crypto++ are as follows ([2]):
   a. GCM, CCM, EAX
b. AES (Rijndael), RC6, MARS, CAST-256, Twofish, Serpent

c. Panama, Sosemanuk, Salsa20, Xsalsa20

d. IDEA, Triple-DES, Camellia, SEAL, Skipjack, SHACAL-2, RC5, Blowfish

e. ECB, CBC, CTS, CFB, OFB, CTR

f. VMAC, HMAC, CBC-MAC, GMAC, TwoTrack-MAC

g. SHA-1, SHA-2, SHA-3, WHIRLPOOL, Tiger, RIPEMD-128, RIPEMD-256, RIPEMD-160, RIPEMD-320

h. ECDSA, ECNR, ECIES, ECMQV, ECDH

i. MD2, MD4, MD5, Panama Hash, Square, GOST, SAFER, SEAL 3.0, DES, ARC4, DESX, RC2, 3-WAY, WAKE-OFB, CAST-128, SHARK

j. Diffie-Hellman, XTR-DH, DH2, MQV, LUCDF

k. PKCS#1 v2.0, OAEP, PSS, IEEE P1363 EMSA2-EMSA5, PSSR

l. ESIGN, LUC, RSA, DSA, ElGamal, RW, NR, DLES

iii. Libmcrypt: The “libmcrypt” cryptography library provides encryption of data and is thread safe. This specific library contains a set of encryption algorithms and modes which are modular in nature. This nature allows algorithms and the encryption modes to operate in a much efficient manner. The various algorithms contained within the framework of this library are tabulated in Table 1:

```
| Algorithm  | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
|-----------|-------|---------|---------|-------|----------|---------|
| IDEA       | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
| TDES       | TDES  | TDES    | TDES    | TDES  | TDES     | TDES    |
| RSA        | RSA   | RSA     | RSA     | RSA   | RSA      | RSA     |
| CBC        | CBC   | CBC     | CBC     | CBC   | CBC      | CBC     |
| CFB        | CFB   | CFB     | CFB     | CFB   | CFB      | CFB     |
| OFB        | OFB   | OFB     | OFB     | OFB   | OFB      | OFB     |
| CTR        | CTR   | CTR     | CTR     | CTR   | CTR      | CTR     |
```

Table 1: Algorithms in Libmcrypt library

iv. Botan (formerly known as OpenCL): This cryptography library is written in C++ and licensed under BSD-2 ((23), (28)). It was later implemented with a “Card Verifiable Certificate” for ePassports and this modified version of Botan was named “InSiTO”. This library contains a number of encryption formats, algorithms and protocols which are tabulated in Table 2:

```
| Algorithm  | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
|-----------|-------|---------|---------|-------|----------|---------|
| IDEA       | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
| TDES       | TDES  | TDES    | TDES    | TDES  | TDES     | TDES    |
| RSA        | RSA   | RSA     | RSA     | RSA   | RSA      | RSA     |
| CBC        | CBC   | CBC     | CBC     | CBC   | CBC      | CBC     |
| CFB        | CFB   | CFB     | CFB     | CFB   | CFB      | CFB     |
| OFB        | OFB   | OFB     | OFB     | OFB   | OFB      | OFB     |
| CTR        | CTR   | CTR     | CTR     | CTR   | CTR      | CTR     |
```

Table 2: Algorithms in Botan library

v. Libgcrypt: Written in C language, the “libgcrypt” is a multi-platform cryptography library licensed under GNU Lesser General Public License GNU General Public License ((32)). It features a multiple precision arithmetic implementation and entropy gathering utility ((37)). The cryptography algorithms in this library are tabulated in Table 3:

```
| Algorithm  | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
|-----------|-------|---------|---------|-------|----------|---------|
| IDEA       | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
| TDES       | TDES  | TDES    | TDES    | TDES  | TDES     | TDES    |
| RSA        | RSA   | RSA     | RSA     | RSA   | RSA      | RSA     |
| CBC        | CBC   | CBC     | CBC     | CBC   | CBC      | CBC     |
| CFB        | CFB   | CFB     | CFB     | CFB   | CFB      | CFB     |
| OFB        | OFB   | OFB     | OFB     | OFB   | OFB      | OFB     |
| CTR        | CTR   | CTR     | CTR     | CTR   | CTR      | CTR     |
```

Table 3: Algorithms in Libgcrypt library

vi. Bouncy Castle: This particular cryptography library is written in Java and C# ((41)). Designed mainly for use in devices with low computational memory, this library contains the algorithms listed in Table 4:

```
| Algorithm  | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
|-----------|-------|---------|---------|-------|----------|---------|
| IDEA       | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
| TDES       | TDES  | TDES    | TDES    | TDES  | TDES     | TDES    |
| RSA        | RSA   | RSA     | RSA     | RSA   | RSA      | RSA     |
| CBC        | CBC   | CBC     | CBC     | CBC   | CBC      | CBC     |
| CFB        | CFB   | CFB     | CFB     | CFB   | CFB      | CFB     |
| OFB        | OFB   | OFB     | OFB     | OFB   | OFB      | OFB     |
| CTR        | CTR   | CTR     | CTR     | CTR   | CTR      | CTR     |
```

Table 4: Algorithms in Bouncy Castle library

vii. Cryptlib: The “cryptlib” cryptography library is a library of algorithms which provides security to communication and information exchange. Its simple interface makes it very user-friendly and its layered structure (the lower layers each providing a layer of abstraction, the higher layers covering up the details of implementation of the algorithms) makes up the whole library very secure and impermeable to intrusion to a very high degree. The various algorithms within this library are tabulated in Table 5:

```
| Algorithm  | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
|-----------|-------|---------|---------|-------|----------|---------|
| IDEA       | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
| TDES       | TDES  | TDES    | TDES    | TDES  | TDES     | TDES    |
| RSA        | RSA   | RSA     | RSA     | RSA   | RSA      | RSA     |
| CBC        | CBC   | CBC     | CBC     | CBC   | CBC      | CBC     |
| CFB        | CFB   | CFB     | CFB     | CFB   | CFB      | CFB     |
| OFB        | OFB   | OFB     | OFB     | OFB   | OFB      | OFB     |
| CTR        | CTR   | CTR     | CTR     | CTR   | CTR      | CTR     |
```

Table 5: Algorithms in Cryptlib library

viii. Catacomb: Written using gcc, this cryptography library contains a set of cryptographic primitives and used in Linux operating systems (9). Some of the most prominent categories of algorithms within this library out of its many other are as shown in Table 6:

```
| Algorithm  | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
|-----------|-------|---------|---------|-------|----------|---------|
| IDEA       | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
| TDES       | TDES  | TDES    | TDES    | TDES  | TDES     | TDES    |
| RSA        | RSA   | RSA     | RSA     | RSA   | RSA      | RSA     |
| CBC        | CBC   | CBC     | CBC     | CBC   | CBC      | CBC     |
| CFB        | CFB   | CFB     | CFB     | CFB   | CFB      | CFB     |
| OFB        | OFB   | OFB     | OFB     | OFB   | OFB      | OFB     |
| CTR        | CTR   | CTR     | CTR     | CTR   | CTR      | CTR     |
```

Table 6: Categories of algorithms in the Catacomb library

ix. Cryptix: The “Cryptix” (say Cx) cryptography library was made to provide a library of cryptographic algorithms to the Java platform as there were a number of issues regarding adoption of cryptography in Java ((22)). With the removal of export controls on cryptography, the use of “Cryptix” (last active development was in 2005) declined with the increasing availability of other more secure cryptography libraries. The list of algorithms under this library are shown in Table 7:

```
| Algorithm  | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
|-----------|-------|---------|---------|-------|----------|---------|
| IDEA       | 3DES  | AES 256 | AES 128 | CAST5 | BLOWFISH | AES 256 |
| TDES       | TDES  | TDES    | TDES    | TDES  | TDES     | TDES    |
| RSA        | RSA   | RSA     | RSA     | RSA   | RSA      | RSA     |
| CBC        | CBC   | CBC     | CBC     | CBC   | CBC      | CBC     |
| CFB        | CFB   | CFB     | CFB     | CFB   | CFB      | CFB     |
| OFB        | OFB   | OFB     | OFB     | OFB   | OFB      | OFB     |
| CTR        | CTR   | CTR     | CTR     | CTR   | CTR      | CTR     |
```

Table 7: Algorithms in Cryptix library
Cryptography Architecture) ([39]). This encryption toolkit is supported by CoreProvider (which contains algorithms like PKCS #1, 3DES etc.), ECPProvider (which contains algorithms based on elliptic curve such ECDH key agreement scheme, ECDSA etc.), PQCProvider (Contains the McEliece cryptosystem in four variants (CFS signature scheme etc.) and NFFProvider (contains IQRDSA, IQDSA, IQGQ etc.).

xi. LibTomCrypt: The “LibTomCrypt” cryptography library is an open source library of cryptographic primitives ([20]).

xii. MatrixSSL: The “MatrixSSL” cryptography library is designed for devices and application with smaller footprint. An implementation of embedded SSL and TLS, it contains various symmetric key and public key algorithms. Some popular algorithms included in this library are given in Table 8:

| Algorithm | AES-GCM | SHA256 | SHA512 |
|-----------|---------|--------|--------|
| RSA       | Diffe-Hellman | Elliptic Curve | AES |
| AES-CCM | SHA256 | BLAKE2-SHA256 |

Table 8: Algorithms in MatrixSSL

xiii. MIRACL: Multiprecision Integer and Rational Arithmetic C Library (MIRACL) is a cryptography library designed for use in constrained environment in terms of size and computational power ([38]).

xiv. Mozilla’s NSS: NSS (Network Security Services) cryptography library facilitates the encryption in server-based applications. It mainly supports the following security algorithms listed in Table 9 for use in server applications:

| Algorithm | SSL | TLS | PKCS #1
|-----------|-----|-----|------|
| RSA       | MD2 | MD4 | MD5 |
| Diffie-Hellman | SHA1 | SHA2-256 |
| AES       | SHA-512 |
| CBC       | AES-128 |
| CCM       | AES-256 |
| ECP       | SHA-256 |
| HMAC      | SHA-256 |

Table 9: Security algorithms in NSS

xv. OpenPGP: This cryptography library is an open source variant of PGP (Pretty Good Privacy) which is used for securing the privacy of end-users and levelling up the security of communication systems by implementation of authentication methods through the use of PGP ([16], [18]).

xvi. OpenSSL: Written in C language, the “OpenSSL” is a multi-platform library of cryptographic algorithms and functions ([40]). It is an open source library licensed under Apache License 1.0 and 4-clause BSD License. It implements the various SSL protocols and TLS protocols. The various algorithms implemented by “OpenSSL” are tabulated in Table 10 as follows:

| Algorithm | DES | RC2 | RC4 | RC5 |
|-----------|-----|-----|-----|-----|
| AES       | SHA-512 |
| RSA       | SHA-512 |
| SHA-512  | SHA-512 |

Table 10: Algorithms implemented by OpenSSL

xvii. Nettle: This is a low-level, multi-platform cryptography library licensed under GNU Lesser General Public License ([17]). The various algorithms within this cryptography library are shown in Table 11:

| Algorithm | AES-CCM | SHA256 |
|-----------|---------|--------|
| RSA       | Diffe-Hellman | Elliptic Curve |
| AES-GCM  | SHA-512 | BLAKE2-SHA256 |

Table 11: Algorithms in Nettle

4. CRYPTOGRAPHY LIBRARIES IN IoT

4.1 WolfSSL (formerly known as CyaSSL): Written in ANSI C, the “wolfSSL” cryptography library, due to its small footprint size and low runtime memory, is aimed to be used in embedded devices, RTOS and environments facing constraints in computational resources ([30], [33]). This library supports the development of cross-platform algorithms and houses a large number of algorithms. Moreover it features the generation of Key and Certificates. “wolfSSL” is licensed under GNU General Public License GPLv2.

“wolfSSL” contains the following categories of algorithms to be used for cryptographic purposes which are shown in Table 12:

| Algorithm | RSA | Diffie-Hellman | SHA-1 | SHA-256 |
|-----------|-----|----------------|-------|---------|
| AES-GCM   | SHA-256 |
| RC4       | SHA-512 |
| RC5       | SHA-512 |
| AES-CCM   | SHA-512 |
| SHA256    | SHA-512 |

Table 12: Algorithm library of wolfSSL

4.2 AvrCryptoLib: Licensed under GPLv3, the “Avr-Crypto-Lib” cryptography library has the implementation of its encryption algorithms in the AVR 8-bit microcontrollers ([34]). As with all the rest of the cryptography library aimed to be used in the field of IoT, the “Avr-Crypto-Lib” is optimized for resource-constrained environments in regards to available computational memory and size.

“Avr-Crypto-Lib” contains vast number algorithms which are categorised in Table 13:

| Algorithm | AES | SHA-1 | SHA-256 |
|-----------|-----|-------|---------|
| SHA256    | SHA-512 |
| RC4       | SHA-512 |
| RC5       | SHA-512 |
| AES-GCM   | SHA-512 |
| SHA512    | SHA-512 |

Table 13: Algorithm library of Avr-Crypto-Lib

Besides the above mentioned algorithms, “Avr-Crypto-Lib” also provides MAC functions and Pseudo Random Number Generators (PRNGs).

4.3 WiseLib: Written in C++, the “Wiselib” cryptography library is targeted to be used in networked embedded devices ([12], [35]). Using “Wiselib”, an individual can compile algorithms for various platforms like Contiki, iSense, Shawn (a simulator of sensor network) etc. using its various in-house algorithms like routing algorithms,
localization algorithms etc. The use of template similar to Boost and CGAL facilitates highly efficient compilations of the various generic and platform independent codes written for various platforms.

4.4 TinyECC: TinyECC is an Elliptic Curve Cryptography based library which can perform ECC-based PKC operations ([25], [29]). Some of the most prominent features of TinyECC are:

a. Provision of ECDSA, a digital signature
b. ECIES, a scheme for encryption of public key
c. ECDH, a protocol for key exchange

5.4 RelicToolKit: Licensed under LGPL v2.1 (and above), the “RELIC toolkit” cryptography library is an efficient and flexible meta-toolkit ([21]). The main use of the “RELIC toolkit” is in its ability to be used for construction of custom cryptographic toolkits.

The various algorithms implemented by the “RELIC toolkit” are as follows:

- Multiple-precision integer arithmetic
- Bilinear maps and extensions fields relate to bilinear maps
- Elliptic curves:
  - Over prime fields
  - Over binary fields
- Prime and Binary field arithmetic
- Cryptographic protocols

The various cryptographic protocols implemented by RELIC are tabulated in Table 14:

| Protocol | RELIC |
|----------|-------|
| ECDSA    |       |
| ECIES    |       |
| ECIES    |       |
| ECIES    |       |

Table 14: Cryptographic protocols of RELIC toolkit

5. COMPARATIVE ANALYSIS OF THE CRYPTOGRAPHY LIBRARIES IN IoT

5.1 WolfSSL: WolfSSL includes OpenSSL compatibility layer along with support for OCSP and CRL which are used for validating certificates. Its runtime memory usage is between 1-36 kB. Sporting a very simple API, this library supports zlib compression, IPv4 and IPv6 along with integration of MySQL ([30]).

5.2 AvrCryptoLib: This cryptography library performs modular exponentiation using C-interfaces in AVR 8-bit assembly language. This leads to reduction in execution time of this cryptography library. Moreover this library allows direct access to keys through storage of these keys in the flash memory which results in efficient consumption of SRAM.

5.3 WiseLib: Implementing elliptic curve over prime fields only, this library shuns away from incorporation of optimizations of assembly level for making its codes platform independent.

5.4 TinyECC: Though mainly made for running in devices operating on TinyOS, TinyECC can be implemented in devices other than TinyOS-dependent devices as the library can be ported to C99 through manual alteration of code parts or through the usage of tool-chains. This library also implements curves over prime fields only and includes sliding windows ([43]) and Barrett reduction ([44]) for the purpose of verification.

5.5 RelicToolKit: The inclusion of multiple integer arithmetic makes its compilation of this library easy for a wide variety of platforms. RelicToolKit provides high level of customization in terms of:

5.5.1 Building and inclusion of desired components only for usage in desired platforms.

5.5.2 Desired selection of various mathematical optimizations for optimum performance of the toolkit in a specific platform.

6. CONCLUSION

From the above comparative analysis of the various features available in the different cryptography library the foremost conclusion is that not one of the cryptography libraries in the IoT environment can be considered as a universal library due to their varying features and optimizations made for different specific platforms. This results in the non-existence of a single universal standard library that can be applicable to all IoT devices around us. Moreover each library contains a specific set of features unique to them and optimized for the platform where these are applicable. And the use of the above cryptography libraries along with the adoption of various security measures that can be adopted in various communication modes ([24], [42]) and implementation of intrusion detection systems and schemes ([19], [27], [31]) will lead to a more secure and reliable IoT infrastructure for wide adoption of its devices by the masses.

7. REFERENCES

[1] Thomas Eisenbarth, Sandeep Kumar, Christof Paar, Axel Poschmann, and Leif Uhsadel. "A survey of lightweight-cryptography implementations". IEEE Design & Test of Computers 24, no. 6 (2007): 522-533.
[2] Dai, Wei. "Crypto++ Library 5.6.0." [http://www.cryptopp.com] (2009).
[3] Sandipan Dey, Hameed Al-Qaheri, and Sugata Sanyal. "Embedding Secret Data in Html Web Page." arXiv preprint arXiv:1004.0459 (2010).
[4] [http://www.sourceforge.org/Programming/Libraries/Networking/Opcode/osiIcmp-1.0.2.tar.gz.shtml]
[5] [http://www.homeport.org/~adam/crypto/]
[6] Whitfield Diffie, and Martin E. Hellman. "New directions in cryptography". Information Theory, IEEE Transactions on 22, no. 6 (1976): 644-654.
[7] Sandipan Dey, Ajith Abraham, Bijoy Bandypadhyay, and Sugata Sanyal. "Data Hiding Techniques Using Prime and Natural Numbers." arXiv preprint arXiv:1003.3672 (2010).
[8] Schneier, Bruce. Applied cryptography: protocols, algorithms, and source code in C. John wiley & sons, 2007.
[9] [http://teal.gmu.edu/courses/Crypto_resources/web_resources/libraries.htm]
[10] Victor Miller. "Use of elliptic curves in cryptography," in *Advances in Cryptology—CRYPTO’85 Proceedings*, pp. 417-426. Springer Berlin/Heidelberg, 1986.

[11] Sugata, Ayu Tiwari, and Sudip Sanyal. "A multifactor secure authentication system for wireless payment." In *Emergent Web Intelligence: Advanced Information Retrieval*, pp. 341-369. Springer London, 2010.

[12] Baumgartner, Tobias, Ioannis Chatzigiannakis, Sándor Fekete, Christos Koninis, Alexander Kröller, and Apostolos Fylgesis. "Wiselib: A generic algorithm library for heterogeneous sensor networks." In *Wireless Sensor Networks*, pp. 162-177. Springer Berlin Heidelberg, 2010.

[13] http://en.wikipedia.org/wiki/Crypto%2B%2B

[14] Vivek Kapoor, Vivek Sonny Abraham and Ramesh Singh. "Elliptic curve cryptography." *ACM Ubiquity* 9, no. 20 (2008): 20-26.

[15] Sandipan Dey, Ajith Abraham, and Sugata Sanyal. "An LSB Data Hiding Technique Using Natural Number Decomposition", in *Intelligent Information Hiding and Multimedia Signal Processing, 2007. IHMSP 2007. Third International Conference on*, vol. 2, pp. 473-476. IEEE, 2007.

[16] Alexander Ulrich, Ralph Holz, Peter Hauck, and Georg Carle. "Investigating the openpgp web of trust", in *Computer Security—ESORICS 2011*, pp. 489-507. Springer Berlin Heidelberg, 2011.

[17] http://en.wikipedia.org/wiki/Nettle_%28cryptographic_library%29

[18] http://www.openpnp.org/

[19] Animesh Kr Trivedi, Rajan Arora, Rishi Kapoor, Sudip Sanyal, and Sugata Sanyal. "A Semi-distributed Reputation Based Intrusion Detection System for Mobile Adhoc Networks." *arXiv preprint arXiv:1006.1956* (2010).

[20] Denis, Tom St. "LibTomCrypt," available online at libtom.org (2004).

[21] https://code.google.com/p/relic-toolkit/

[22] Gayoso Martinez, V., C. Sanchez Avila, J. Espinosa Garcia, and L. Hernandez Encinas. "Elliptic curve cryptography: Java implementation issues." In *Security Technology, 2005. CCST’05. 39th Annual 2005 International Carnahan Conference on*, pp. 238-241. IEEE, 2005.

[23] http://resources.infosecinstitute.com/cryptographic-libraries/

[24] Sugata Sanyal, Rohit Bhadauria, and Chittabrata Ghosh. "Secure communication in cognitive radio networks," in *Computers and Devices for Communication, 2009. CODEC 2009. 4th International Conference on*, pp. 1-4. IEEE, 2009.

[25] http://discovery.csc.ncsu.edu/software/TinyECC/

[26] Muhammad Yasir Malik. "Efficient implementation of elliptic curve cryptography using low-power digital signal processor", in *Advanced Communication Technology (ICACT), 2010 The 12th International Conference on*, vol. 2, pp. 1464-1468. IEEE, 2010.

[27] Tapalina Bhattachasi, Rituparna Chaki, and Sugata Sanyal. "Sleep deprivation attack detection in wireless sensor network". *arXiv preprint arXiv:1203.0231* (2012).

[28] Lloyd, Jack. "Botan Cryptographic Library," h ttp://botan.randombit.net.