Energy Analysis of Wireless Sensor Networks using RSA and ECC Encryption Method

Amanjot Kaur

ABSTRACT

Wireless Sensor Networks consist of sensor nodes and few powerful control mobile laptops performing activities like routing, data aggregation etc over wireless media. However such kind of environment prone great security threats due to the broadcast nature of the transmission medium. Sensor networks pose unique challenges; traditional security techniques used in traditional networks cannot be applied directly. Therefore this paper investigates cryptography algorithms to showcase energy analysis. The work is implemented over a network created with the help of Java Sun SPOT kit, consisting of sensor node and base station.

Keywords
Wireless Sensor Networks, Elliptic curve Cryptography, Sun Spots, Energy Analysis.

---

1. Wireless Sensor Network

Sensor networks refer to a heterogeneous system combining tiny sensors and actuators with general-purpose computing elements. These networks will consist of hundreds or thousands of self-organizing, low-power, low-cost wireless nodes deployed to monitor and affect the environment [1]. Sensor networks are typically characterized by limited power supplies, low bandwidth, small memory sizes and limited energy. This leads to a very demanding environment to provide security.

2. Security in WSN

Wireless networks are vulnerable to security attacks due to the broadcast nature of the transmission medium. Furthermore, wireless sensor networks have an additional vulnerability because nodes are often placed in a hostile or dangerous environment where they are not physically protected.

2.1 Wireless Sensor Network Security Challenges

Because sensor networks pose unique challenges, traditional security techniques used in traditional networks cannot be applied directly. First, to make sensor networks economically profitable as sensor devices are limited in their energy, computation, and communication capabilities. Second, sensor nodes are often deployed in accessible areas, presenting the added risk of physical attack [2]. And third, sensor networks interact closely with their physical environments and with people, posing new security problems. Five of the most important challenges are described below.

Wireless Medium: Wireless medium is less secure because its broadcast nature makes eavesdropping simple. Any transmission can easily be intercepted, altered, or replayed by an adversary. It allows an attacker to easily intercept valid packets and easily inject malicious ones.

Ad-Hoc Deployment: Network topology keeps changing due to node failure, addition & mobility. So nothing is known of the topology prior to deployment. Security schemes require robust designs to cope and operate in dynamic and ever changing environment.

Hostile Environment: The highly hostile environment represents a serious challenge. Attackers can capture a node, physically disassemble it, and extract from it valuable information.

Resource Limitation: Energy is the most precious resource for sensor networks.. Security mechanisms must be energy efficient otherwise it could increase its cost and also make decrease its usability.

Big Scale Network: The high scale of sensor networks poses a significant challenge for security mechanisms. Providing security for it is equally challenging. Security mechanisms must be scalable to very large networks maintaining high computation and communication efficiency.

2.2 Security Requirements in Wireless Sensor Network

Major security issues for Wireless Sensor Network are listed below.

Data Confidentiality: Confidentiality means keeping information secret from unauthorized parties. A sensor network should not leak sensor readings to neighbouring networks. In many applications nodes communicate highly sensitive data. Simple method to keep sensitive data secret is to encrypt the data with a secret key that only legitimate receivers possess. Public-key cryptography is little expensive to be used in the resource constrained sensor networks.

Data Authenticity and Integrity: One can easily inject malicious data in sensor network. So receiver should make sure that data it received is correct and legitimate data as in this malicious data can lead to wrong interpretation by receiver which can have very bad impact as in critical decision making process wrong information can lead to catastrophic results also.

Data Freshness: Data freshness implies that the data is recent, and it ensures no old messages are replayed over network. For this counter must be used that can determine freshness.

Availability: Availability ensures that services and information can be accessed at the time they are required. In sensor networks there are many risks that could result in loss of availability such as sensor node capturing and denial of service attacks.

3. Cryptography

Cryptography is basically the conversion of data into a secret code for transmission over a public network. Today's cryptography is more than encryption and decryption. Cryptography is the study of "mathematical" systems for solving two kinds of security problems: privacy and authentication [3].

3.1 Private Key Cryptography

A single key is used for both encryption and decryption. Encryption the data is encrypted by any encryption algorithm using the key. Only the user having the access to the same 'key' can decrypt the encrypted data. Examples are AES, 3DES etc.

3.2 Public Key Cryptography

In public key cryptography each user or the device taking part in the communication have a pair of keys, a public key and a private key, and a set of operations associated with the keys to do the cryptographic operations . Only the particular device knows the private key whereas the public key is distributed to all devices taking part in the communication. Examples are RSA, ECC etc.

3.2.1 RSA Algorithm

Key generation:

1. Select random prime numbers p and q, and check that p != q
2. Compute modulus n = pq
3. Compute phi, ϕ= (p - 1)(q - 1)
4. Select public exponent *e*, 1 < e < ϕ such that gcd(e, ϕ) = 1
5. Compute private exponent d = e^{-1} mod ϕ
6. Public key is {n, e}, private key is d

Encryption: $c = m^e \bmod n$, decryption: $m = c^d \bmod n$

Digital signature: $s = H(m)^d \bmod n$, Verification: $m' = s^e \bmod n$, if m' = H(m) signature is correct. H is a publicly known hash function.

3.2.2 ECC Algorithm

Elliptic curves were proposed for use as the basis for discrete logarithm-based cryptosystems almost 25 years ago independently by Victor Miller [4] of IBM and Neal Koblitz [5] of the University of Washington. Elliptic key operates on smaller key size. ECC operates on the points in the elliptic curve $y^2=x^3+ax+b$, where $4a^3+27b^2\neq 0$. Equation of elliptic curve is in real coordinate. In prime field operation the elliptic curve equation is modified as:

$y^2 \bmod p = x^3+ax+b \bmod p$, where $4a^3+27b^2 \bmod p \neq 0$.

An elliptic curve over a finite field F_P is composed of a finite group of points (x_i, y_i) where integer coordinates x_i, y_i satisfy the long Weierstrass Equation.

$y^3 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$

And the coefficients a_1, a_2, a_3, a_4, a_6 are elements of F_P and are the parameters of the curve. The curve discriminent is $\Delta\neq 0$ and there is also a point at infinity denoted by Θ. If F_P is a field of characteristics 2, then the curve is called a binary elliptic curve. Since the field is F_P is generally used is cryptographic applications, the equation of elliptic curve is simplified to [20]

$y^2 = x^3 + ax + b$, $a,b \in Fp$, $4a^3 + 27b^2 \neq 0$

The requirement $4a^3 + 27b^2 \neq 0$ ensures that E is non-singular, this means in particular that one may compute the tangent in every point on the curve.

The set of rational points in E over Fp denoted by E(Fp) is

$E(Fp) = \{(x, y) \in Fp^2 : y^2 = x^3 + ax + b\} \cup \{O\}$,

where O is the point at infinity. The elements of $E(F_P)$ are called points of elliptic curve. The public key is a point in the curve and the private key is a random number.

3.3 Limitations of Algorithms implemented over WSN

Sensor networks are vulnerable to resource consumption attacks. Intruder can repeatedly send packets to drain the nodes batteries and waste network bandwidth. Computing power, memory, and battery life of devices are more constrained in WSN and all above security protocols use lots of memory and require lot of battery power to successfully implement them. Implementing them in wireless Sensor network actually degrades performance of WSN due to large battery and memory required by them to implement. Elliptical curve cryptography provides same level of security with less parameters required thus improving the performance of Wireless Sensor Network.

3.4 Key Comparison of RSA and ECC

RSA most commonly used public-key cryptosystem. As due to advancement and increase in computing power available to an adversary, both symmetric and public key sizes must grow over time so that acceptable security for a fixed protection life span can be provided and Table 1 shows this expected key-size growth for various symmetric and public-key cryptosystems.

Table 1 Computationally equivalent key sizes [6]

Symmetric	ECC	RSA/DSA
80	163	1024
128	283	3072
192	409	7680
256	571	15360

4. Experimental Setup

The entire implementation is done using Java Sun Spot kit. Java program (Client) running on Sunspot send data using secure connection via base station to server running on computer. Sunspot only understand 3 cipher suits - TLS_ECDH_ECDSA_WITH_RC4_128_SHA, SSL_RSA_WITH_RC4_128_SHA, SSL_RSA_WITH_RC4_128_MD5

Free Sun SPOT sends data to server which is running on host machine, via base station. Base station is connected to server via USB cable. Data is collected in separate text files for each algorithms. Figure 1 shows source code of program running on Sunspot and Figure 2 shows source code of Server used during implementation.

```
public class SensorSampler extends MIDlet {
  private static final int HOST_PORT = 1000;
  private static final int SAMPLE_PERIOD = 27*1000;  // in milliseconds

  protected void startApp() throws MIDletStateChangeException {
    StreamConnection conn;
    DataOutputStream outSPOT=null;
    String ourAddress = System.getProperty("IEEE_ADDRESS");
    ISCalarInput lightSensor =  EDemoBoard.getInstance().getLightSensor();
    IBattery battery = Spot.getInstance().getPowerController().getBattery();
    long now;
    int reading = 0;
    double batt;
    int i;
    ITriColorLED[] leds = EDemoBoard.getInstance().getLEDs();
    try {

      conn = (StreamConnection)Connector.open("sslsocket://172.31.28.113:" + HOST_PORT,
        Connector.READ_WRITE, true);
     outSPOT = new DataOutputStream(conn.openOutputStream());
    }
    catch (Exception e) {
      System.err.println("Caught " + e + "in connection initialization.");
      System.exit(1);
    }
    //System.out.println("Starting sensing action and then send it to Server ... \n");
    while(true) {
      try {
        batt= battery.getAvailableCapacity() ;
        now = System.currentTimeMillis();
        reading = lightSensor.getValue();
        for(i=0;i<=7;i++)
        {
        leds[i].setRGB(255, 255, 255);
        leds[i].setOn();
        }
         outSPOT.writeInt(reading);
         outSPOT.writeDouble(batt);
         outSPOT.writeLong(now);
         outSPOT.flush();
        for (i=0;i<=7;i++)
        leds[i].setOff();
        Utils.sleep(SAMPLE_PERIOD - (System.currentTimeMillis() - now));
      }
```

Figure 1 Java Code running on Sun SPOT.

```
Serverv10.java - Notepad
File  Edit  Format  View  Help
import java.io.InputStream;
import java.io.DataInputStream;
import java.io.BufferedInputStream;
import java.net.ServerSocket;
import java.net.Socket;
import javax.net.ServerSocketFactory;
import javax.net.ssl.SSLServerSocketFactory;
import javax.net.ssl.SSLServerSocket;

public class Serverv10 {
      port in which the server is listening for connections
      private final static int PORT = 1000;

      public static void main (String[] argv) throws Exception {
      SSLServerSocketFactory ssocketFactory = (SSLServerSocketFactory)SSLServerSocketFactory.getD
      SSLServerSocket ssocket = (SSLServerSocket)ssocketFactory.createServerSocket(PORT);

      System.out.println("Server running ... \n");

      String[] enabledCipherSuites = { "SSL_RSA_WITH_RC4_128_SHA"};
      ssocket.setEnabledCipherSuites(enabledCipherSuites);

      System.out.println("Server waiting for client ... \n");

      Socket socket = ssocket.accept();
      DataInputStream input=new DataInputStream(socket.getInputStream());
      System.out.println("Server connected to a client ... \n");

      while(true) {
         DataInputStream input=new DataInputStream(new BufferedInputStream(socket.getInputStream(
         int vall = input.readInt();
         double val = input.readDouble();
         long tim = input.readLong();
         System.out.println("Light reading is " + vall + "  Battery Level is " + val + "  Time is
      }
    }
  }

//TLS_ECDH_ECDSA_WITH_RC4_128_SHA
//SSL_RSA_WITH_RC4_128_SHA
```

Depending on which cipher suit to use value of enabledCipherSuits will change

Figure 2 Server Code running on host machine

ECC and RSA certificates are created and these certificates are added into trust store of sunspot and server. In this we are getting light reading, battery capacity available and time from sunspot and it is sending these readings to server and it store result in text file.

4.1 Results

As per the literature Sun Spot takes around 3-4 hours to charge it fully via USB. Here in this experiment Sun Spot is charged for 4 hours and then run once with RSA algorithm and next time with ECC algorithm and find that battery sustain longer time when data is encrypted using ECC algorithm than RSA algorithm. As already mentioned data collected is light reading, battery value and time (in milliseconds). The time mentioned is in milliseconds which is calculated as total milliseconds from 1 January 1970, 00:00 GMT till that time (it is calculated as number of years*no of days in each year*number of hours in each day *number of minutes in each hour * number of seconds in each minute * 1000). To calculate the total time Sun Spot take to discharge whole battery, difference of last reading and first reading is taken. Table 2 shows first and last reading using RSA algorithm and Table 3 shows first and last reading using ECC algorithm.

Table 2 Output recorded using RSA

Reading No	Light Reading	Available Battery	Time (in milliseconds)
First	1	163.95456944444445	1335603586528
Last	2	16.0	1335610487738

Table 3 Output recorded using ECC

Reading No	Light Reading	Available Battery	Time (in milliseconds)
First	4	163.86642361111112	1335505476994
Last	9	16.0	1335512917294

Under same circumstances, from above readings it is observed that by using RSA algorithm for encryption battery discharges in 6901210 milliseconds (1335610487738-1335603586528) i.e approx. 115.020 minutes or 1.917 hrs. By using ECC algorithm for encryption battery discharges in 7440300 milliseconds (1335505476994- 1335505476994) i.e. approx. 124.005 minutes or 2.066. Experiment is ran 10 times for each algorithm and it is observed that battery consumption is less when ECC algorithm is used to encrypt data as compared to RSA.

It was observed that battery level never reaches zero rather it never drops beyond reading 16. In Sun Spot the battery level is computed based on the measured discharge rate and so is approximate. It is also computed conservatively which is why it goes to 16 after about few hrs and stays there until the battery voltage really falls at the end.

Its to be kept in mind that Sun Spot will run for longer duration if one increase its sleep time. Actually for most percentage of time Sun Spot is sleeping and only for few milliseconds of time it actually operates. Changing lights of Sun Spot also saves energy. Different LED lights take different amount of energy. Sun Spot can run for 7-12 hours depending upon its sleep time, Led lights used. But under same circumstances when encoded data is transferred ECC algorithm takes less battery energy than RSA algorithm as experimentally proved above.

5. CONCLUSION

This paper designed and implemented RSA and ECC for establishing secure connection in wireless sensor networks. At lower key bit size ECC provide same level of security as RSA. Implemented the algorithms one by one over Sun SPOTs for analyzing the cryptographic behaviour. Here Sun SPOT send the light reading , available battery capacity and time.

First charged the battery and then calculated the time taken by each algorithm to discharge the battery to find out which algorithm takes less amount of battery power or consumes less battery. Under same circumstances RSA consumes more battery power as compared to ECC.

6. FUTURE SCOPE

In future this research work can be extended as during secure transfer of data calculate the memory consumed by RSA and ECC algorithms. Calculate time taken to transfer the data between secure connection by each algorithm to find which takes less bandwidth to transfer data. Design and implement a set of attacks against ECC.

7. REFERNCES

[1] Matt Welsh, Dan Myung, Mark Gaynor, and Steve Moulton "Resuscitation monitoring with a wireless sensor network", in Supplement to Circulation: Journal of the American Heart Association, October 2003.

[2] Adrian Perrig, John Stankovic, David Wagner, "Security in Wireless Sensor Networks", in Communications of The ACM, Vol 47, No.6, June 2004.

[3] W. Diffie & M. Hellman, "New directions in cryptography", IEEE Transaction on Information Theory, IT-22(6), pp. 644-654, 1976.

[4] V. Miller, "Use of elliptic curves in cryptography, Advances in Cryptology − CRYPTO 85", Springer-Verlag, pp. 417-426, 1985.

[5] N. Koblitz, "Elliptic curve cryptosystems", Mathematics of Computation, pp 203-209, 1987.

[6] Vipul Gupta, Sumit Gupta, Sheueling Chang, "Performance Analysis of Elliptic Curve Cryptography for SSL" Performance analysis of elliptic curve cryptography for SSL. In *Proc. 3rd ACM Workshop on Wireless Security*, pp. 87–94. ACM Press, 2002

Amanjot Kaur
Assistant Professor
S.D.S.P.M.College For Women, Rayya(Amritsar)