A secure IoT based flood warning system using elliptical curve cryptography

Ummer Iqbal Khan1*, D.S. Oberoi1
1 National Institute of Electronics and Information Technology, Srinagar/Jammu, India

Abstract

Objectives: To propose a design of an early flood warning system based on IoT with a strong emphasis on the security requirements of such systems.

Methods/Analysis: The design of an early flood warning system is based on measuring the hydrological parameters of a river, which include water level and discharge through an IoT based network. To calculate the discharge of water in a river, Manning's equation has been employed. Security protocols based on elliptical curve cryptography has been proposed for authenticating the real-time hydrological data. The proposed security protocols have been formally validated and verified against various active and passive attacks using AVISPA and Scyther. Simulation of the proposed scheme has also been carried out on the TOSSIM simulator using the TinyECC library to estimate the radio and CPU energy overheads.

Findings: The formal security validation of the proposed authentication scheme indicates that the scheme is SAFE against various active and passive attacks. The simulation on Scyther indicates that no attacks have been found in proposed authentication protocols. The AVISPA validation also declares the scheme as SAFE as all-important security goals have been achieved. The Energy analyses on TOSSIM indicate that each node in the scheme requires 133 mJ for radio transmission and 59 mJ for CPU operations about the Mica2 Energy Model.

Novelty/Improvement: Researchers have proposed many designs and schemes for early flood warnings without giving much emphasis to the security of such systems. The paper focuses on the security requirements of such systems.

Keywords: IoT; flood warning; authentication; ECC; AVISPA; Scyther; TinyOS; TinyECC; TOSSIM

1 Introduction

Scientists across the world are constantly developing technologies to monitor the flowing rivers to avert the destruction caused by it. The key hydrological parameters like water discharge, water level, and precipitation provide an insight into the behavior of the river like floods, spill, and discharge.
These parameters can thus be used to implement early warnings, spill detection and continuous discharge. The traditional approach for monitoring hydrological is primarily based on spot and grab techniques. These techniques are less accurate and have a high financial burden. Standardized techniques for the measurement and analysis of hydrological data, primarily through means of modern communication technology, are essential for understanding and predicting the magnitude and frequency of floods, the change in the course of the river, and the water quality parameters of a river etc (1). The need to have real-time monitored data such as flow rate, precipitation, water level, is essential in order to make a reasonable decision on the actions necessary to be performed to allow rivers to continue benefiting humans and prevent rivers from destroying human investments (2).

Internet of Things (IoT) and Wireless Sensor Networks (WSN) presents a promising solution for developing new flood warnings systems. The design of such systems primarily involves interfacing of hydrological sensors with data acquisition boards fitted with the Mote (3). The sensed data is then transported to the IoT gateway through a multihop mesh network. However, the authentication of hydrological data is of paramount importance as, based on this data, flood warnings are disseminated (4).

The resource constraint nature of IoT/WSN almost pre-emptes the use of conventional Public Key Cryptography schemes and digital signatures based on RSA. ECC has shown more promise for the application of asymmetric techniques for authentication in WSN. ECC (5) can achieve the same level of security as RSA with smaller key sizes, e.g., 160 Bit ECC can provide comparable security to the conventional 1024 Bit RSA (6). Smaller key size often brings the advantage of faster computation efficiency and saving of bandwidth, memory and energy. That makes ECC better suited for resource constraint devices like WSN/IoT.

This study proposed a secure flood warning system designed in an IoT environment. It enabled a lightweight security framework to secure the communication pattern involved in the system. The Security solution leverages the low computational overheads associated with Elliptical Curve Cryptography (ECC) without using any digital signature algorithm. The proposed security protocols have been formally verified by Avispa (7) and Scyther (8). The scheme has been simulated on the TOSSIM simulator, and a relevant discussion based on the performance of the proposed security scheme is also presented.

1.1 Related work

In (9) proposed a flood prediction system using Wireless Sensor Networks. The system is based on linear regression, which uses a polynomial to predict a rise in floodwater. In (10) presented a real-time flood monitoring system based on low power devices. The system was used in southern Spain. In (9) proposed a flood warning system based which transmitted the data on GPRS from a remote location. In (11) proposed a flood warning system using empirical formulae. Samman (12) presented a simulation model based on numerical techniques for flood prediction in Saudi Arabia.

In (13) proposed a Real-Time Flash-Flood Monitoring, alerting, and forecasting system using data mining and wireless sensor network. The system includes the variables, which include temperature, humidity, and vibration measured through wireless sensor nodes. (14–18) presented early warning system for water flooding based on simplified parameters. In (19,20) proposed visual sensing mechanism for early flood warning. The visual sensing method for urban flood monitoring is used in (18,19) as a solution for an early warning system. Visual sensors provided a real-time picture of affected sites.

The existing designs in the literature have not addressed the security aspects of a real-time early flood warning system. The proposed work primarily focuses on the security aspects of such systems. As most of the early flood warning systems comprise of low power devices, conventional security mechanisms cannot be employed as such. Typically the security need of such systems involves authentication of communicated hydrological data. If the sensor data is not authenticated, a masqueraded data can subvert a false alarm. The paper focuses on elliptical curve cryptography to provide algorithms for data authentication.
2 Preliminaries

2.1 Elliptical curve cryptography

An elliptical Curve satisfies the equation (1):
\[ y^2 = x^3 + ax + b \]  

In the elliptical curve group \((E,+)\) defined over a finite field \(F_p\) for some prime no ‘P’ let \(G(x,y)\) be a generator point that can generate every other point in the group. When this point \(G(x,y)\) is added ‘n’ number of times to itself, the point addition yields another point \(Q(x,y)\) belonging to the same Elliptical curve.

\[ Q(x,y) = n \cdot G(x,y) \quad \text{where } n \text{ is scalar multiplication operation} \]  

Given \(G(x,y)\) and \(Q(x,y)\) are known, finding ‘n’ is computationally infeasible and is called an Elliptical Curve Discrete Log Problem(ECDLP). Choosing a suitably large Field makes it computationally more challenging to find ‘n’.

2.2 TinyOS

TinyOS (21) is an open-source, event-based operating system designed to specifically meet the requirements of resource constraint networks like IoT and WSN. It features a component-based architecture that enables rapid implementation with limited code size. Its Execution model is similar to a Finite State Machine (FSM) with more programmable options. Its event-based ability and split phase operation allow a high degree of concurrency to be handled in limited memory space. TOSSIM is the simulator used in TinyOS.

2.3 TinyECC

TinyECC (22) provides a simple, configurable, flexible, and ready-to-use software library for developing WSN/IOT based applications on TinyOS. All the ECC operations, including point addition, point doubling, and point multiplications, are supported by TinyECC.

3 Proposed secure flood warning system

The architecture of the proposed system is built around a host of sensors that are interfaced with a data acquisition system and sensor network, as shown in Figure 1. A set of Hydrological sensors are deployed at various strategically important locations (Sensor spots). These Sensor spots are deployed across the length of the river at various critical conjunctions. The sensors placed at the sensor-spots are interfaced with a data-acquisition system using a wired connection. The various hydrological parameters to be monitored include Discharge, Water Level, Precipitation (Rainfall).

Discharge is the volume of water moving down a stream or river per unit of time, commonly expressed in cubic feet per second or gallons per day. River discharge monitoring can play an important role in ascertaining the possibility of floods. Discharge can be calculated using Manning’s equation. For using Manning’s equation, an initial survey of the cross-section of the river is done to establish the topography of the river. The free surface slope of the water is determined by measuring the level of water at several points along the length of the river for which level sensors can be employed. The calculation of instantaneous discharge is based on (2):

\[ Q = \frac{1}{n} AR^{2/3}S^{1/2} \]  

Where \(n\) is the Manning’s coefficient of rugosity, a constant depending on the characteristics of the place to be measured, \(A\) is the area of flow (in the cross-section of the river), \(R\) is the hydraulic radius computed as Area over the wetted perimeter and \(S\) is the slope. Manning equation is the ISO 1070:1992(E) standard.
The schematics to compute the discharge using the manning equation are shown in Figure 2. The Data acquisition nodes behaving as an RFD's in an IoT environment acquire the water level reading from the various chosen location. Through a multihop based routing protocol, these readings are relayed to the gateway. The gateway further relays the level reading over a point to point link to a community center where a slope, as well as discharge, is calculated. The sensors used for measurement generate a current output in the range of 4 to 19 mA proportional to the parameters being measured. The current output from different sensors is fed to the ADC channel of a data acquisition board like MDA300\textsuperscript{(23)}. Data acquisition board MDA 300 supports up to 8 channels of 16-bit analog input with single-ended 0 to 2.5-volt inputs or 4 differential 0 to 2.5 volt ADC channel. It also supports 8 digital 0 to 2.5 volt IO channels, 64K EEPROM for sensor calibration data, 200Hz counter channel, and external 12C interface. The data acquisition boards fitted with a wireless sensor network mote would transmit the sensor readings to a gateway. The sensing and transmission of data are done by an embedded program written in NesC language based on TinyOS\textsuperscript{(20)} operating system.
The program intelligently senses the parameters by dynamically changing the sampling rate and selective data transmission for effective management of power and data storage. The Gateway is used to bridge wired and wireless components of the network and shall be programmed as a coordinator that receives the data sent from the sensor nodes wirelessly. An application server is connected to gateway using either Ethernet or USB interface. Community centers are equipped with necessary hardware and software's which includes a computer system and actuators for alarming the local population about various anomalies. The computer system runs an instance of a serial forwarder, which populates the global database. The data dispatched from various community center’s will finally populate a global database through an Internet cloud. As an enormous amount of data is being generated over some time, effective data management can be employed for efficient data storage and retrieval using big data concepts.

4 System security for WSN based flood warning system

Security is one of the most imperative aspects of any system. From an architectural perspective, the proposed early flood system can be divided into 3 tiers: Mote tier, Server Tier, Client Tier. Mote Tier involves the sensor network infrastructure for sensing and forwarding the hydrological parameters. The server tier involves the Calibration of raw data from the mote tier. It also provides the functionality of a serial forwarder. Client tier involves data visualization and analysis. The security of data in the client and server tier can be handled by a conventional cryptographic method. However, the security of the Mote tier is a challenge as it involves low power devices for which conventional cryptographic methods are not potent.

Hydrological data is generally meant for public viewing; hence there is no need for any confidentiality. The risk of injecting false packets (Sybil Attack)\(^\text{(21)}\) by an adversary into the network would exist. This can result in disruption of flood monitoring by raising false alarms or subverting a genuine alarm. Thus the fundamental security requirement for the proposed system is authentication of data. Authentication is used to establish that the communication pattern between Node and Gateway is legitimate.

In Mote Tier, RFD is responsible for sensing Hydrological parameters and transmitting these readings to the cluster FFD or directly to the Gateway depending upon the topology being used. In either case, it becomes imperative to authenticate the data from the RFD as based upon these values, certain actuations or alarms may be triggered. On the other hand, a user query needs to be forwarded from Gateway to the respective RFD for processing. Thus it becomes quite essential to provide authentication in the following cases assuming a flat topology:

- Gateway to RFD Communication
- RFD to Gateway Communications

4.1 Authentication protocols based on ECC

We present a set of lightweight protocols for achieving authenticated RFD to Gateway, Gateway to RFD Communication. The proposed authentication protocols are based on elliptical curve cryptography, thus making it suitable for resource constraint devices\(^\text{(24–26)}\). The Notations used in the framework are tabulated in Table 1. During the initialization, RFD selects a private key \(X_A\) and computes the public key as \(P_A(x,y) = X_A.G(x,y)\), and Gateway selects a private essential \(S_B\) and computes the public key as \(P_{BASE}(x,y) = S_B.G(x,y)\).

4.1.1 Gateway to RFD Authentication

In this authentication algorithm, an RFD\(_A\) authenticates Gateway.

**Step 1**: RFD\(_A\) Computes \(M_1(x,y) = ID_A * P_A(x,y)\) and sends it to the Gateway

**Step 2**: Gateway Computes \(M_2(x,y) = M_1(x,y) * S_B\) and Sends it to the RFD\(_A\)

**Step 3**: RFD\(_A\) Computes \(C_K(x,y) = ID_A * X_A * P_{BASE}(x,y)\)

**Step 4**: RFD\(_A\) authenticates Gateway if \(M_2(x,y) == C_K(x,y)\)
Table 1. Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFD</td>
<td>Reduced Function Device</td>
</tr>
<tr>
<td>$X_A$</td>
<td>Private Key of RFD A</td>
</tr>
<tr>
<td>$S_B$</td>
<td>Private Key of MIB600</td>
</tr>
<tr>
<td>$P_A(x,y)$</td>
<td>Public Key of RFD A</td>
</tr>
<tr>
<td>$P_{BASE}(x,y)$</td>
<td>Public Key of Gateway</td>
</tr>
<tr>
<td>$G(x,y)$</td>
<td>Generator Point of Elliptical Curve</td>
</tr>
<tr>
<td>$H()$</td>
<td>Hash Function</td>
</tr>
<tr>
<td>$ID_A$</td>
<td>Identity of RFD A</td>
</tr>
<tr>
<td>+</td>
<td>Point Addition</td>
</tr>
<tr>
<td>*</td>
<td>Scalar Multiplication</td>
</tr>
</tbody>
</table>

4.1.2 RFD to Gateway Authentication
In this authentication algorithm, Gateway authenticates $RFD_A$.

**Step 1**: $RFD_A$ calculates $H(ID_A)$ and computes $M_3(x,y) = [H(ID_A) + X_A] \times G(x,y)$

**Step 2**: $RFD_A$ sends $M_3(x,y), ID_A$ to the Gateway

**Step 3**: Gateway stores $[M_3(x,y), ID_A]$ and computes $D_K(x,y) = [H(ID_A) \times G(x,y) + P_A(x,y)]$

**Step 4**: Gateway checks if $(M_3(x,y) == D_K(x,y))$, if true then $RFD_A$ is authenticated.

5 Formal Verification using AVISPA and Scyther
Avispa is an automated validation tool for security protocols. It is a push-button tool based on Dolev and Yao\(^{(27)}\) model. Dolev and Yao\(^{(28)}\) attack model gives complete control of the communication channel to the intruder. The intruder, in this case, can forward, modify, and change messages but cannot overuse the computational strength of an algorithm. In Avispa, the protocols are modeled using HLPSL. It is a role-based formal language that comprises of roles compositions, security models, etc. The architecture of Avispa is shown in **Figure 3**.

![Avispa Architecture](https://www.indjst.org/)

The protocol implemented in HLPSL is translated into IF format using an HLPSL to OF translator. The IF format is then passed into various AVISPA backend, which includes OFMC, ATSE, SATMC, and TA4SP. These backends check a protocol against active and passive attacks. The HLPSL Scripts for RFD to Gateway and Gateway to RFD are

https://www.indjst.org/
shown in Figure 4 and Figure 5.

The OFMC outputs are shown in Figure 6 and Figure 7. The outputs indicate that the proposed protocols are safe against active and passive attacks.

Scyther is an automated security protocol validation tool developed by Cremers. In Scyther, security protocols are modeled in Scyther Protocol Description Language (SPDL). In SPDL, communicating parties are modeled as roles, and the communication pattern is specified within these roles. The communication between the specified roles is implemented using send and recv operations. In order to verify the security strength of a protocol, various types of claims are declared within the roles defining the protocol. In Scyther, there are 6 types of claims: 1. Secret: 2. Session-Key-Reveal (SKR): Weak Agree 4. Alive. 4. NI_Agree: 4. NI_Synch. The verification of the protocol in scyther is indicated in Figure 8. From Figure 8, it can be depicted that the protocol is verified against all major claims depicting that protocol is safe against various active and passive attacks.

```lisp
Role role_A(A:agent, B:agent, G:token, H:channel, SND, RCV:channel(dy))
  played_by A
  def =
    local
    init
    State := 0

  transition
  1. State = 0 \& RCV(Pka) = [\rightarrow State = 1 \& Na := new() \& PKa' := (G, Na, SND, PKa) \& secret(Na', private_A, A)]
  2. State = 1 \& RCV(Pkb) = [\rightarrow State = 2 \& Nid := new() \& En' := (H, PKb, Nid) \& SND(En') \& request(A, mib_to_rfd, En')]
  4. State = 2 \& RCV(Em') = [\rightarrow State = 3]

end role

role role_B(A:agent, B:agent, G:token, H:channel, SND, RCV:channel(dy))
  played_by B
  def =
    local
    init
    State := 0

  transition
  1. State = 0 \& RCV(Pkb) = [\rightarrow State = 1 \& Nb := new() \& PKb' := (H, Nb, SND, PKb) \& secret(Nb', private_B, B)]
  4. State = 1 \& RCV(En) = [\rightarrow State = 2 \& Em := (En', Nb) \& SND(En') \& witness(B, A, mib_to_rfd, Ev)]

end role

role session(A:agent, B:agent, G:token)(U:function)
  def =
    local
    SND, RCV, SND1, RCV1:channel(dy)
    composition
    role_B(A, B, G, H, SND, RCV2) \& role_A(A, B, G, SND, RCV1)

end role

role environment()
  def =
    const
    bob.Agent, function, alice.Agent, g:token, ni.text, sec_dhvalue : protocol_id, private_A : protocol_id, private_B : protocol_id, mib_to_rfd, protocol_id, Intruder_knowledge = (alice, bob, g)
    composition
    session(alice, bob, g) \& session(alice, g, h) \& session(alice, i, g, h)

end role

goal
  secrecy_ofsec_dhvalue
  secrecy_ofprivate_A
  secrecy_ofprivate_B
  authentication_mib_to_rfd
end goal
environment()
```

Fig 4. HLPSL script for gateway to RFD
Fig 5. HLPSL Script for RFD to gateway
Fig 6. OFMC output for gateway to RFD

Fig 7. OFMC output for RFD to gateway
The proposed design and scheme has been implemented on TinyOS using TinyECC Library. An IoT/WSN application in TinyOS is component-based and supports split-phase operations. Component-based development involves developing applications in a modular way, thus supporting modularity. Split phase operations, also called an asynchronous method, calls help in effective duty cycle management, thus conserving energy. A component in TinyOS is implemented through an interface. Accessing components through interfaces helps in standardization in user/system component development. An IoT application in TinyOS is graphically represented using a component graph. The component graph graphically depicts the usage of various components that are used in developing the application, as well as the interface service provided and used by a component. The component graph of TinyECC is shown in Figure 9. The component graph of the developed application is shown the Figure 10.
The developed application on TinyOS has been simulated on TinyViz, a graphical user interface of the TOSSIM Simulator. A snapshot of TinyViz simulation is shown in Figure 11. The Total RAM and ROM consumed are shown in Figure 12. The energy consumed by the proposed scheme has been computed using PowerTOSSIM. PowerTOSSIM is an energy estimation module of TOSSIM. The energy module used in calculating the CPU and Radio power consumption by PowerTOSSIM is depicted in Table 2. The average energy consumed by a node for implementing the proposed scheme is shown in Figure 13.
Fig 12. Memory consumed

Table 2. Energy model

<table>
<thead>
<tr>
<th>State</th>
<th>CPU</th>
<th>Radio</th>
<th>LED/Sensor Board/EEPROM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>8.0 mA</td>
<td>Rx 7.0 mA</td>
<td>Led's 6.2 mA</td>
</tr>
<tr>
<td>Idle</td>
<td>3.2 mA</td>
<td>Tx(-20 dBm) 3.7 mA</td>
<td>Sensor Board 0.7 mA</td>
</tr>
<tr>
<td>ADC Noise Reduce</td>
<td>1.0 mA</td>
<td>Tx(-19 dBm) 5.2 mA</td>
<td>EEPROM</td>
</tr>
<tr>
<td>Power Down</td>
<td>103 μA</td>
<td>Tx(-15 dBm) 5.4 mA</td>
<td>Read 6.2mA</td>
</tr>
<tr>
<td>Power Save</td>
<td>110 μA</td>
<td>Tx(-8 dBm) 6.5 mA</td>
<td>Read Time 565 μs</td>
</tr>
<tr>
<td>Stand By</td>
<td>216 μA</td>
<td>Tx(-5 dBm) 7.1 mA</td>
<td></td>
</tr>
<tr>
<td>Extended Standby</td>
<td>223 μA</td>
<td>Tx(0 dBm) 8.5 mA</td>
<td>Write 18.4 mA</td>
</tr>
<tr>
<td>Internal Oscillator</td>
<td>0.93 μA</td>
<td>Tx(+4 dBm) 11.6 mA</td>
<td>Write Time 12.9 ms</td>
</tr>
</tbody>
</table>

Fig 13. Energy Consumed per Node as depicted by PowerTOSSIM
7 Conclusion

In this study, a secured system for early flood warning using wireless sensor networks has been presented. Many Schemes have been given by various researchers for using IoT/WSN for early flood warnings. However, not much work has been done in securing such systems. A lightweight security framework has been presented to achieve authentication between Gateway and RFD devices used in the system. The developed protocols were formally verified using AVISPA and Scyther. The formal analysis depicted that the protocol was found safe against various active and passive attacks. The protocol has also been implemented on TinyOS and simulated on the TOSSIM platform. The energy overheads of the scheme have also been calculated.

References


https://www.indjst.org/