INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL

Check for updates

OPEN ACCESS

Received: 31-03-2020
Accepted: 31-05-2020
Published: 22-06-2020

Editor: Dr Natarajan Gajendran

Citation: Khan UI, Oberoi DS (2020) **A secure IoT based flood warning system using elliptical curve cryptography**. Indian Journal of Science and Technology 13(21): 2080-2093. https://doi.org/10.17485/IJST/v13i21.73

*Corresponding author.
Ummer Iqbal Khan
National Institute of Electronics and Information Technology, Srinagar/Jammu, India
Ummer.iqbal.khan@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

A secure IoT based flood warning system using elliptical curve cryptography

Ummer Iqbal Khan[1]*, **D.S. Oberoi**[1]

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 dissemaited [(4)].

The resource constraint nature of IoT/WSN almost pre-empties 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 addresed 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):

$$y^2 = x^3 + ax + b \quad (1)$$

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.

$$Q(x,y) = n.\,G(x,y) \quad \textit{where . is scalar multiplication operation} \quad (2)$$

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):

$$Q = 1/n\,AR^{2/3}S^{1/2} \quad (2)$$

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.

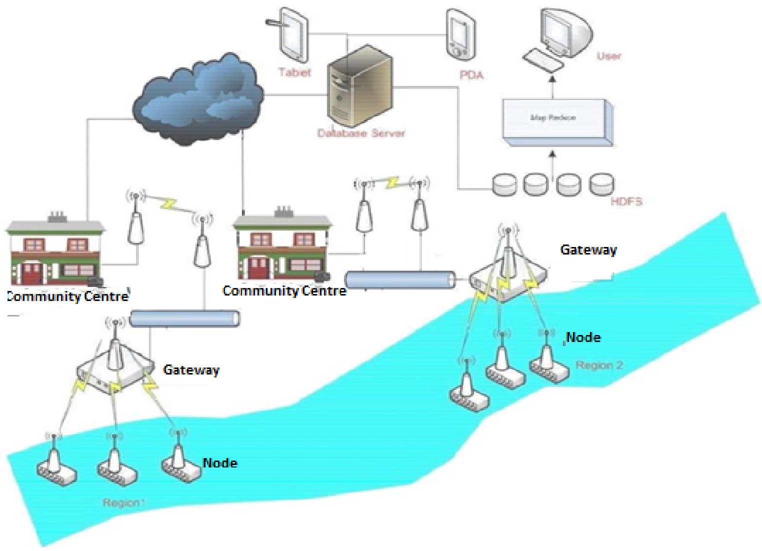


Fig 1. Basic schematic of the proposed system

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 [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 [20] operating system.

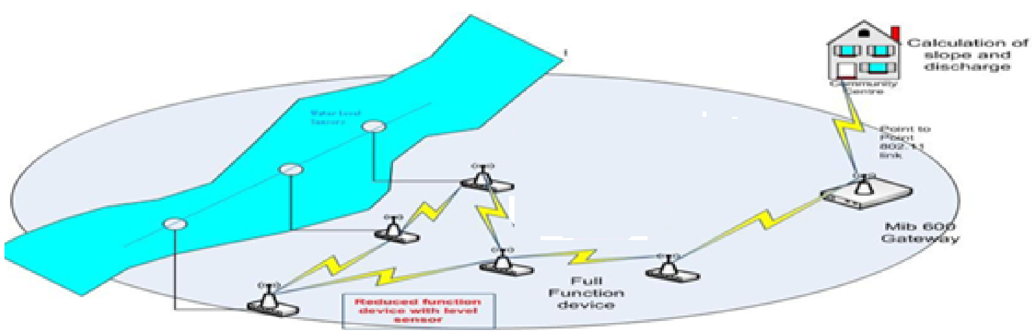


Fig 2. Discharge calculation

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)[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[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

Symbol	Description
RFD	Reduced Function Device
X_A	Private Key of RFD A
S_B	Private Key of MIB600
P_A(x,y)	Public Key of RFD A
P_{BASE}(x,y)	Public Key of Gateway
G(x,y)	Generator Point of Elliptical Curve
H()	Hash Function
ID_A	Identity of RFD A
+	Point Addition
*	Scalar Multiplication

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 $M_3(x, y) = [H(ID_A)+ X_A] * G(x,y)$

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 $D_K(x,y) = [H(ID_A) * G(x, y) + P_A(x, y)]$

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 Yoa [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 overdue 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.

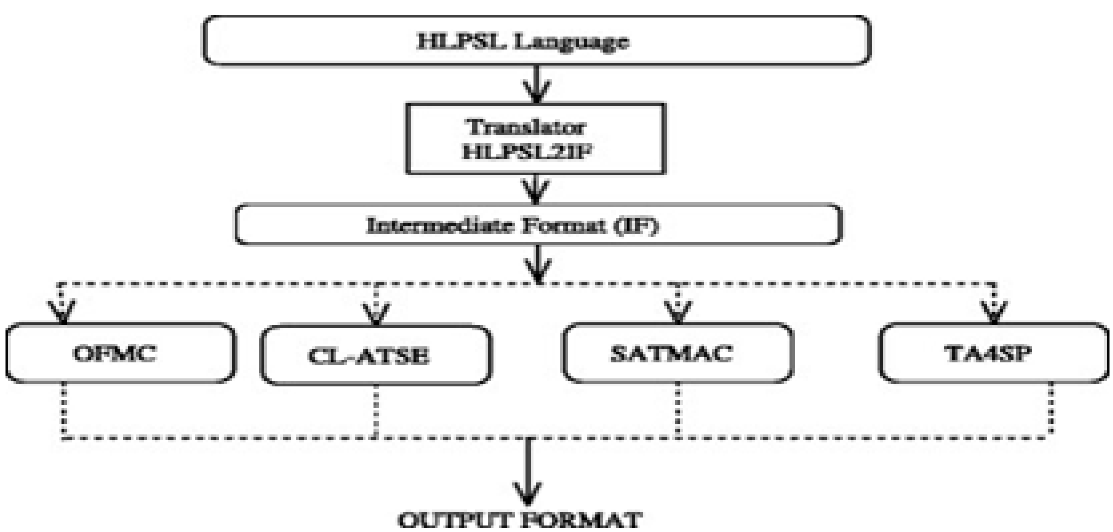


Fig 3. Avispa Architecture

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 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.

```
Role role_A(A:agent,B:agent,G:text,H:function,SND,RCV:channel(dy))
played_by A
def=
            local
State:nat,Nb:text,Na:text,Nid:text,PKa:text, PKb: text , En:text, Em: text
            init
State := 0
            transition
1. State=0 /\ RCV(start) =|> State':=1 /\ Na':=new() /\  PKa':=H(G.Na') /\  SND(PKa') /\ secret (Na' , private_A, {A})
2. State=1 /\ RCV(PKb') =|> State':=2 /\ Nid':=new() /\  En':= H(PKb'.Nid')  /\ SND (En') /\ request(A,B,mib_to_rfd, En')
4. State=2 /\ RCV(Em')  =|> State':=3
end role

role role_B(A:agent,B:agent,G:text,H:function,SND,RCV:channel(dy))
played_by B
def=
            local
State:nat,Nb:text,Na:text,NSecret:text, PKa:text , PKb:text , En: text ,Em:text
            init
                        State := 0
            transition
1. State=0 /\ RCV(PKa') =|> State':=1 /\ Nb':=new() /\ PKb':= H(G.Nb') /\ secret (Nb' , private_B , {B})
3. State=1 /\ RCV(En') =|> State':=2 /\ Em':= H(En'.Nb) /\ SND (Em') /\ witness(B,A,mib_to_rfd,En')
end role

role session(A:agent,B:agent,G:text,H:function)
def=
            local
SND2,RCV2,SND1,RCV1:channel(dy)
            composition
role_B(A,B,G,H,SND2,RCV2) /\ role_A(A,B,G,H,SND1,RCV1)
end role

role environment()
def=
            const
bob:agent,h:function,alice:agent,g:text, ni:text ,sec_dhvalue : protocol_id ,private_A : protocol_id, private_B :
protocol_id , rfd_to_mib: protocol_id
            intruder_knowledge = {alice,bob,g,h}
            composition
session(alice,bob,g,h)/\ session(i,alice,g,h) /\ session(alice, i,g,h)
end role
goal
secrecy_ofsec_dhvalue
secrecy_ofprivate_A
secrecy_ofprivate_B
authentication_onmib_to_rfd
end goal
environment()
```

Fig 4. HLPSL script for gateway to RFD

```
role role_A(A:agent,B:agent,G:text,H:function,SND,RCV:channel(dy))
played_by A
def=
            local
State:nat,Nb:text,Na:text,Nid:text,PKa:text, PKb: text , En:text, Em: text
            init
State := 0
            transition
1. State=0 /\ RCV(start) =|> State':=1 /\ Na':=new() /\ PKa':=H(G.Na') /\ SND(PKa') /\ secret (Na' , private_A, {A})
2. State=1 /\ RCV(PKb') =|> State':=2 /\ Nid':=new() /\ En':= H(xor(H(Nid'),Na).G) /\ SND (En') /\
witness(A,B,rfd_to_mib, En')
 4. State=2 /\ RCV(Em') =|> State':=3
end role


role role_B(A:agent,B:agent,G:text,H:function,SND,RCV:channel(dy))
played_by B
def=
            local
State:nat,Nb:text,Na:text,NSecret:text, PKa:text , PKb:text , En: text ,Em:text
            init
State := 0
            transition
1. State=0 /\ RCV(PKa') =|> State':=1 /\ Nb':=new() /\ PKb':= H(G.Nb') /\ secret (Nb' , private_B , {B})
3. State=1 /\ RCV(En') =|> State':=2 /\ request(B,A,rfd_to_mib,En')
end role


role session(A:agent,B:agent,G:text,H:function)
def=
            local
SND2,RCV2,SND1,RCV1:channel(dy)
            composition
role_B(A,B,G,H,SND2,RCV2) /\ role_A(A,B,G,H,SND1,RCV1)
end role


role environment()
def=
            const
bob:agent,h:function,alice:agent,g:text, ni:text ,sec_dhvalue : protocol_id ,private_A : protocol_id, private_B :
protocol_id , rfd_to_mib: protocol_id
            intruder_knowledge = {alice,bob,g,h}
            composition
session(alice,bob,g,h)/\ session(i,alice,g,h) /\ session(alice, i,g,h)
end role


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

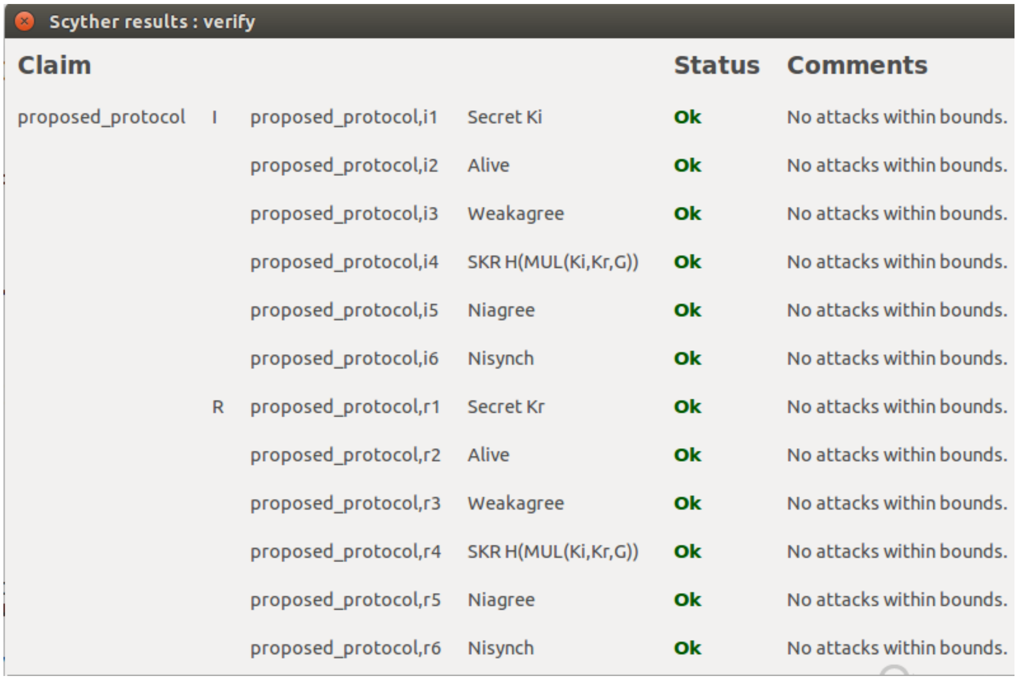
Fig 5. HLPSL Script for RFD to gateway

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/Gateway_to_RFD.if
GOAL
as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime: 0.01s
visitedNodes: 16 nodes
depth: 4 plies
```

Fig 6. OFMC output for gateway to RFD

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/RFD_to_Gateway.if
GOAL
as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime: 0.01s
visitedNodes: 16 nodes
depth: 4 plies
```

Fig 7. OFMC output for RFD to gateway



Scyther results : verify

Claim				Status	Comments
proposed_protocol	I	proposed_protocol,i1	Secret Ki	Ok	No attacks within bounds.
		proposed_protocol,i2	Alive	Ok	No attacks within bounds.
		proposed_protocol,i3	Weakagree	Ok	No attacks within bounds.
		proposed_protocol,i4	SKR H(MUL(Ki,Kr,G))	Ok	No attacks within bounds.
		proposed_protocol,i5	Niagree	Ok	No attacks within bounds.
		proposed_protocol,i6	Nisynch	Ok	No attacks within bounds.
	R	proposed_protocol,r1	Secret Kr	Ok	No attacks within bounds.
		proposed_protocol,r2	Alive	Ok	No attacks within bounds.
		proposed_protocol,r3	Weakagree	Ok	No attacks within bounds.
		proposed_protocol,r4	SKR H(MUL(Ki,Kr,G))	Ok	No attacks within bounds.
		proposed_protocol,r5	Niagree	Ok	No attacks within bounds.
		proposed_protocol,r6	Nisynch	Ok	No attacks within bounds.

Fig 8. Verification result in Scyther

6 Implementation and Simulation

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.

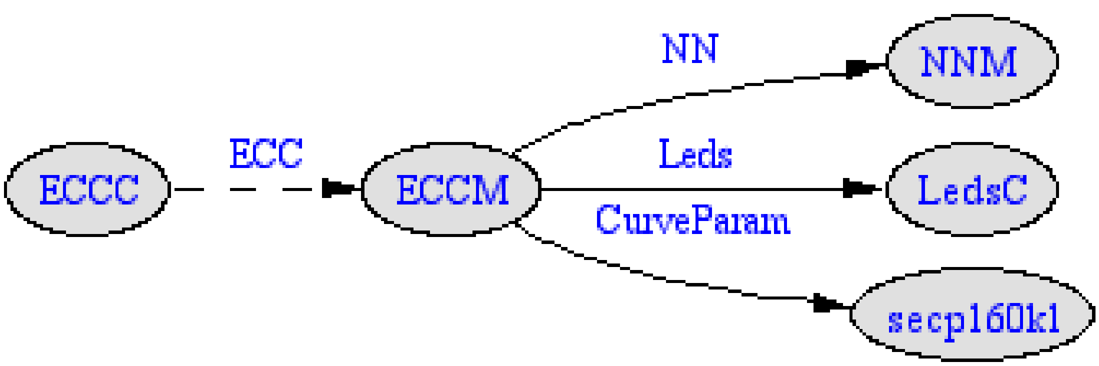


Fig 9. Component Graph of TinyECC

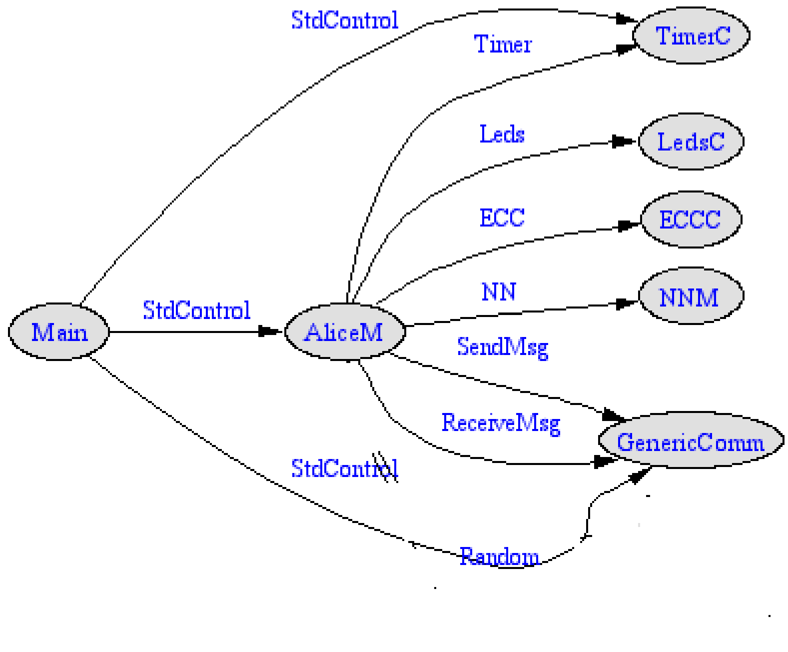


Fig 10. Component Graph of the proposed scheme

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 Power TOSSIM is depicted in Table 2. The average energy consumed by a node for implementing the proposed scheme is shown in Figure 13.

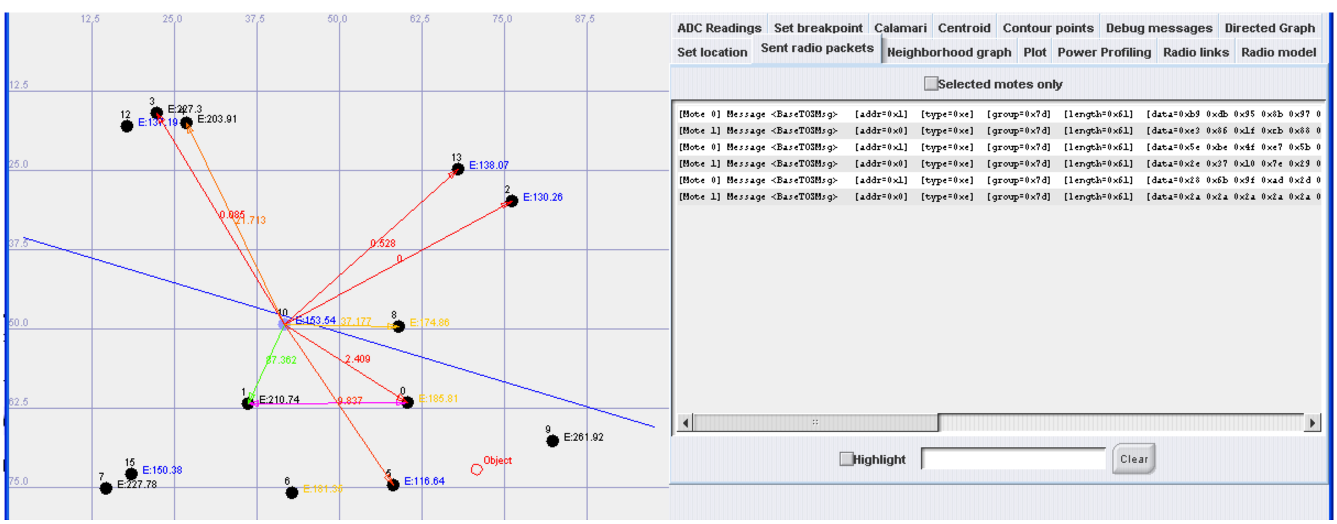


Fig 11. imulation snapshot on TinyViz

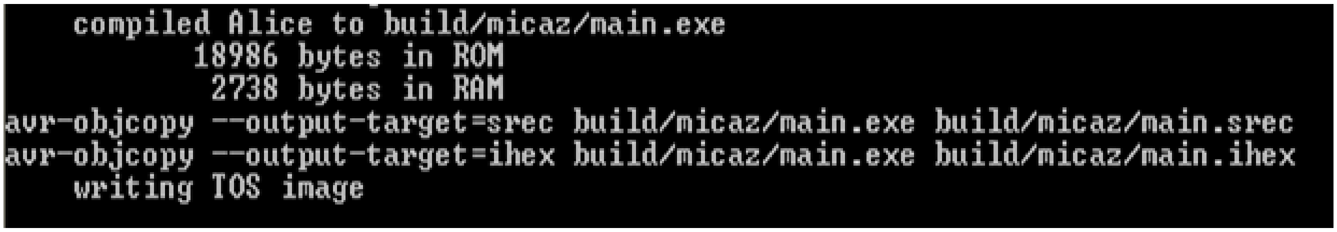


Fig 12. Memory consumed

Table 2. Energy model

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

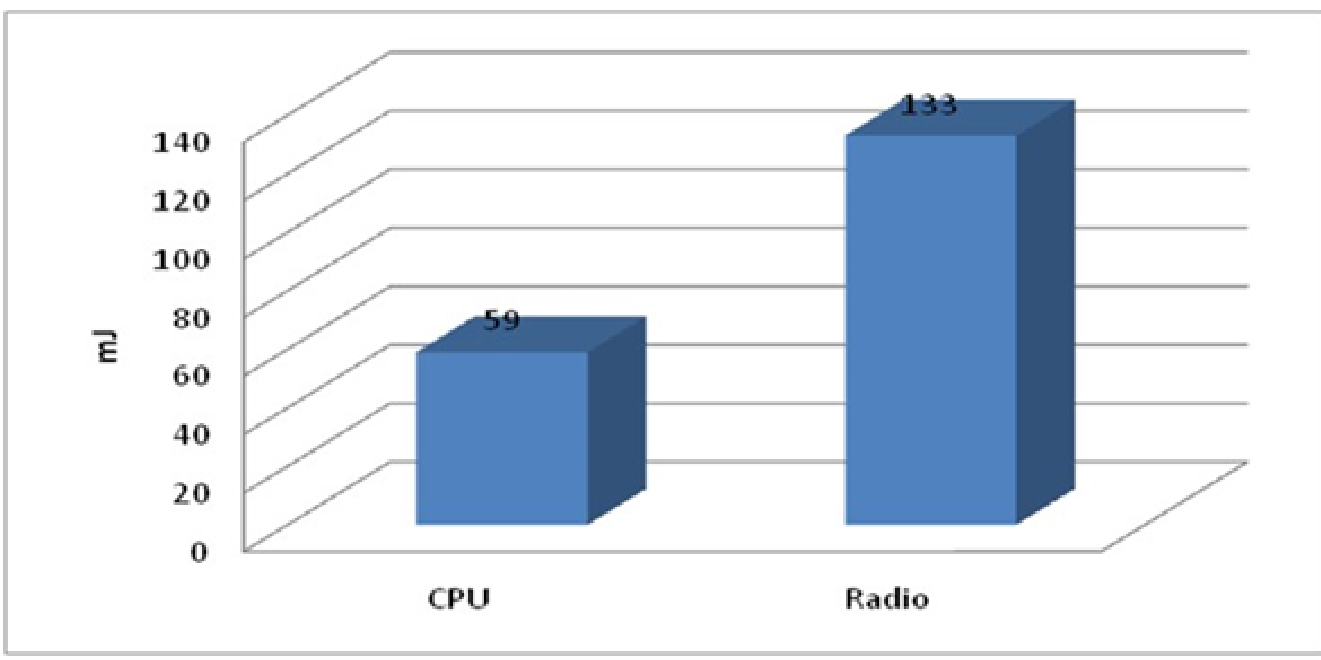


Fig 13. Energy Consumed per Node as depicted by PowerTOSSIM

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

1) Sanchez-Rosario F, Sanchez-Rosario D, Etal JBAH. A Low Consumption Real Time Environmental Monitoring System for Smart Cities based on ZigBee Wireless Sensor Network. In: and others, editor. Proceedings of IEEE International Conference on Wireless Communications and Mobile Computing. 2015.
2) Anees. Applications of Remote Sensing, Hydrology and Geophysics for Flood Analysis. *Indian journal of Science and Technology*. 2017. doi:10.17485/ijst/2017/v10i17/111541.
3) Hattabguesmi. Wireless Smart Sensor Networks for Real-Time Warning System of Flash Floods and Torrents. *KSA International Journal of Computer Applications (CECNet)*. 2017. doi:10.5120/ijca2017913240.
4) Moon AH, Iqbal U, Bhat GM. Secured Data Acquisition System for Smart Water Applications using WSN. *Indian journal of Science and Technology*. 2016. doi:10.17485/ijst/2016/v9i10/86694.
5) Raya M, Hubaux JP. The security of vehicular ad hoc networks. In: CCS05: 12th ACM Conference on Computer and Communications Security 2005. New York. ACM press. 2005;p. 59–64.
6) Gura N, Patel A, Wander A, Eberle H, Shantz SC. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In: Joye M, Quisquater JJ, et al., editors. Cryptographic Hardware and Embedded Systems - CHES 2004 ;vol. 3156. Berlin, Heidelberg. Springer. 2004;p. 119–132. Available from: https://doi.org/10.1007/978-3-540-28632-5_9.
7) AVISPA. AVISPA Web tool: Automated validation of internet security protocols and applications. 2017.
8) Cremers CJF, Computer Aided Verification. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. In: Gupta A, Malik S, editors. Lecture Notes in Computer Science;vol. 5123. Berlin, Germany. Springer. 2008. doi:https://doi.org/10.1007/978-3-540-70545-1_38.
9) Seal V, Raha A, Maity S, Mitra S, Mukherjee A, Kantinaskar M. A simple flood forecasting scheme using wireless sensor networks”. *International Journal of Adhoc, Sensor & Ubiquitous Computing (IJASUC)*. 2012;3(1).
10) Marin-Perez R, García-Pintado J, Gómez AS. A Real-Time Measurement System for Long-Life Flood Monitoring and Warning Applications. *Sensors*. 2012;12(4):4213–4236. Available from: https://dx.doi.org/10.3390/s120404213.
11) Keoduangsine S, Goodwin R. A GPRS-Based Data Collection and Transmission for Flood Warning System: The Case of the Lower Mekong River Basin. *International Journal of Innovation, Management and Technology*. 2012;3(3).
12) Hassoun SAA. Developing an empirical formulae to estimate rainfall intensity in Riyadh region. *Journal of King Saud University - Engineering Sciences*. 2011;23(2):81–88. Available from: https://dx.doi.org/10.1016/j.jksues.2011.03.003.
13) Mane SS, Mokashi MK. Real-Time Flash-Flood Monitoring, Alerting and Forecasting System using Data Mining and Wireless Sensor Network. *IEEE ICCSP*. 2015;p. 1881–1886. Available from: https://doi.org/10.1109/ICCSP.2015.7322851.
14) Dawood MS, Suganya J, Devi RK, Athisha G. A Review on Wireless Sensor Network Protocol for Disaster Management. *International Journal of Computer Applications Technology and Research*. 2013;2(2):141–146. Available from: https://dx.doi.org/10.7753/ijcatr0202.1011.
15) Degrossi, et al. Wireless Sensor Networks for Flood Monitoring in Brazil. In: Proceedings of the 10th International ISCRAM Conference. 2013. Available from: https://doi.org/10.7753/IJCATR0202.1011.
16) Sunkpho J, Ootamakorn C. Real-time flood monitoring and warning system. *Songklanakarin J Sci Technol*. 2011;33(2):227–235.
17) Krzhizhanovskaya VV, Shirshov GS, Melnikova NB, Belleman RG, Rusadi FI, Broekhuijsen BJ, et al. Flood early warning system: design, implementation and computational modules. *Procedia Computer Science*. 2011;4:106–115. Available from: https://dx.doi.org/10.1016/j.procs.2011.04.012.
18) Tejaswitha1 V, Jagadeeshbabu M. Monitoring of Water Level Variations in Rivers and Flood Alert System Using Wireless Sensor Networks. *International Research Journal of Engineering and Technology (IRJET)*. 2016;03(07). Available from: https://doi.org/10.17148/IJARCCE.2015.4885.
19) Lo SW, Wu JH, Lin FP, Hsu CH, Visual Sensing for Urban Flood Monitoring. *Sensors*. 2015;15(8):20006–20029. Available from: https://doi.org/10.3390/s150820006.
20) Elkhrachy I. Flash Flood Hazard Mapping Using Satellite Images and GIS Tools: A case study of Najran City, Kingdom of Saudi Arabia (KSA). *The Egyptian Journal of Remote Sensing and Space Science*. 2015;18:261–278. Available from: https://dx.doi.org/10.1016/j.ejrs.

2015.06.007.
21) Levis P, Gay D, Tinyosprogramming. Tiny OS Programming. and others, editor;Cambridge University Press. 2009.
22) Liu P, Ning, et al. Tiny ECC: A Configurable Library for Elliptical Curve Cryptography in Wireless Sensor Networks. In: and others, editor. 7th International Conference on Information Processing in Sensor Networks SPOTS Track. 2008. Available from: https://doi.org/10.1109/IPSN.2008.47.
23) Memsic. Xserve User Manual. 2007.
24) Hankerson D, Vanstone S, Menezes A. Guide to Elliptic Curve Cryptography. New York. Springer. 2004. Available from: https://doi.org/10.1007/b97644.
25) Dhillon PK, Kalra S. Elliptic curve cryptography for real time embedded systems in IoT networks. In: and others, editor. 5th International Conference on Wireless Networks and Embedded Systems (WECON). 2016. Available from: https://doi.org/10.1109/wecon.2016.7993462.
26) El-hajj M, Fadlallah A, Chamoun M, Serhrouchni A. A Survey of Internet of Things (IoT) Authentication Schemes. *Sensors*. 2019;19(5):1141–1141. Available from: https://dx.doi.org/10.3390/s19051141. doi:10.3390/s19051141.
27) Dolev D, Yao A. On the security of public key protocols. In: Proceedings of the IEEE 22nd Annual Symposium on Foundations of Computer Science. 1981;p. 350–357.
28) Dolev D, Yao A. On the security of public key protocols. *IEEE Transactions on Information Theory*. 1983;29(2):198–208. Available from: https://dx.doi.org/10.1109/tit.1983.1056650.