KEY AUTHENTICATION AND ENCRYPTION USING EX-OR FUNCTION IN WIRELESS SENSOR NETWORKS

A.M.Mohammed raffik, R.Subagangadharan, P.Vasudevan nampoothiri, S.Vignesh

Department of Electronics and communication Engineering, Akshaya College of Engineering and Technology, Coimbatore.

Abstract— Wireless sensor networks (WSNs) have been widely used, most notably in real-time traffic monitoring and military sensing and tracking. However, WSN applications could suffer from threats and endanger the applications if the suitable security issues are not taken into consideration. As a result, user authentication is an important concern to protect data access from unauthorized users. This paper presents a lightweight mutual authentication protocol for WSN applications. Instead of traditionally using a hash function for data protection, one of the interesting aspects of this protocol is that, for the purpose of data protection but with a low computational cost, the proposed key encryption function only requires simple exclusive-OR (XOR) arithmetic operations. Moreover, the corresponding hardware architecture was implemented by using an Altera DE2 board, including an AlteraCyclone II field-programmable gate array (FPGA). Finally, the output waveforms from the FPGA were displayed on the 16702A logic analysis system for real-time verification.

Index Terms—Authentication protocol, security, wireless sensor network (WSN).

I. INTRODUCTION

The sensor nodes are transceivers usually scattered in a sensor field where each of them has the capability to collect data and route data back to the Sink / gateway and the end-users by a multi-hop infrastructure less architecture through the sink. They use their processing capabilities to locally carry out simple computations and transmit only the required and partially processed data. The sink may communicate with the task manager/end-user via the Internet or satellite or any type of wireless network (like WiFi, mesh networks, cellular systems, WiMAX, etc.), making Internet of Things possible. However, in many cases the sink can be directly connected to the end-users. Note that there may be multiple sinks/gateways and multiple end-users in the architecture.

We presented an authentication protocol where users can successfully authenticate with any subset of sensors out of a set of n sensors. Further utilized public key cryptography (PKC) and elliptic curve cryptography (ECC) to design a new authentication mechanism. A distributed entity authentication architecture was introduced. It is established on the self-certified key cryptosystem, which is a modification of ECC. The advanced user authentication scheme proposed by Butun was based on the fact that it employed both the PKC and symmetric key cryptography schemes. This approach provides higher energy efficiency as compared to the existing PKC-based schemes, whereas the PKC or ECC based scheme suffers from a high computational cost for WSNs. To alleviate the computational cost, proposed a dynamic password-based authentication scheme. Although this scheme only requires one-way hash functions and simple XOR operations, it is vulnerable to many attacks such as replay attacks, forgery attacks, and so on. An improved scheme presented by the authors in possesses several advantages, including reduction of password leakage risk, capability of changeable password, and better efficiency. We pointed out the fact that Wong et al.’s scheme is susceptible to stolen verifier attacks and showed that the proposed method, what is called “two-factor user authentication,” can be used to avoid multiple users with the same login-id and stolen-verifier attacks. However, Das scheme has no ability to resist gateway node bypass attack and privileged-insider attack. There is no provision of users to change or update their passwords. Khan–Algahathbar’s scheme was given in [12] for solving these security flaws, but it does not provide a mutual authentication mechanism between the user and the gateway. In pointed out the stolen smart card attacks and forgery attacks with node capture attacks in Khan–Algahathbar scheme.

Consequently, a smart-card-based password user authentication scheme was developed to provide against the various stolen smart card attacks. The mutual authentication between the gateway and the sensor node is also provided. Further improved scheme and presented an efficient two factor user authentication mechanism for WSNs, which is still based on password and smart card. This scheme allows the user to choose and change his password frequently. More recently, several researchers indicated diverse security flaws in a two-factor authentication scheme and came up with their
improved versions. It is clear, however, that most of the aforementioned schemes have not been implemented in hardware.

Wireless sensor networks play an important role in information transmission and have a wide variety of applications such as real-time traffic monitoring, building safety monitoring, military sensing and tracking, and so on. They are composed of many tiny and low-cost sensor nodes with limited energy and computation ability to cooperatively monitor physical environmental information. It is well known that WSNs are highly vulnerable and become a threat, thereby endangering the applications if the suitable security is not taken into account. Therefore, how to secure WSNs has been becoming a challenging issue as it presents a resource-constrained environment. User authentication is one of the most crucial security mechanisms to prevent the illegal or malicious entities from accessing the WSNs. In the past decades, several authentication schemes in WSNs have been proposed.

II. TYPES OF SECURITY ATTACKS

a). Denial of Service

Denial of Service (DoS) [Doucet, 2002], [Perrig et al. 2002] is produced by the unintentional failure of nodes or malicious action. In wireless sensor networks, several types of DoS attacks [Wang&Schulzrinne, 2004] in different layers might be performed. At physical layer the DoS attacks could be jamming and tampering, at link layer, collision, exhaustion, unfairness, at network layer, neglect and greed, homing, misdirection, black holes and at transport layer this attack could be performed by malicious flooding and de-synchronization. The mechanisms to prevent DoS attacks include payment for network resources, pushback, strong authentication and identification of traffic.

b). Sybil Attack

In many cases, the sensors in a wireless sensor network might need to work together to accomplish a task, hence they can use distribution of subtasks and redundancy of information. In such a situation, a node can pretend to be more than one node using the identities of other legitimate nodes (Figure 2.1). This type of attack where a node forges the identities of more than one node is the Sybil attack [Ganesan et al. 2003]. [Karlof& Wagner, 2003]. Sybil attack tries to degrade the integrity of data, security and resource utilization that the distributed algorithm attempts to achieve. Sybil attack can be performed for attacking the distributed storage, routing mechanism, data aggregation, voting, fair resource allocation and misbehavior detection [Karlof& Wagner, 2003].

c). Black hole/Sinkhole Attack

In this attack, a malicious node acts as a black hole [Shayder et al. 2004] to attract all the traffic in the sensor network. Especially in a flooding based protocol, the attacker listens to requests for routes then replies to the target nodes that it contains the high quality or shortest path to the base station. Once the malicious device has been able to insert itself between the communicating nodes, it is able to do anything with the packets passing between them. In fact, this attack can affect even the nodes those are considerably far from the base stations.

d). Hello Flood Attack

Hello Flood Attack [Wang et al. 2008] uses HELLO packets as a weapon to convince the sensors in WSN. In this sort of attack an attacker with a high radio transmission (termed as a laptop-class attacker in [Wang et al. 2008] range) and processing power sends HELLO packets to a number of sensor nodes which are dispersed in a large area within a WSN. The sensors are thus persuaded that the adversary is their neighbour.

e). Wormhole Attack

Wormhole attack [Karlof & Wagner, 2003] is a critical attack in which the attacker records the packets (or bits) at one location in the network and tunnels those to another location. The tunneling or retransmitting of bits could be done selectively. Wormhole attack is a significant threat to wireless sensor networks, because, this sort of attack does not require compromising a sensor in the network rather, it could be performed even at the initial phase when the sensors start to discover the neighboring information.

Now we see the security methods to prevent these attacks. That are achieved by simple Ex-OR operations using Golay encoder.

III. PROPOSED METHOD:

The Key Gen function and algorithm are presented as follows. Let us represent the 32-b message (Msg) and password (PW) in binary (base 2) as

\[ \text{Msg} = m_0m_1m_2\ldots m_{31} \ldots (1) \]
\[ \text{PW} = p_0p_1p_2\ldots p_{31} \ldots (2) \]
The 32-b random number Rx is represented by
\[ Rx = RxM | RxL \quad \ldots \ldots \ldots \ldots \ldots (3) \]
Where,
RxM and RxL are denoted as 16 most significant bits (MSBs) and 16 least significant bits (LSBs), respectively. \(|\) denotes the bitwise concatenation operation. Let RxM and RxL be
\[ RxM = d1d2d3d4 \ldots \ldots \ldots \ldots \ldots (4) \]
\[ RxL = ds1ds2ds3ds4 \ldots \ldots \ldots \ldots \ldots (5) \]
in hexadecimal (base 16), respectively. Each digit of RxM and RxL is used to indicate a bit location in Msg and concatenates these bits to form a 16bit output in hexadecimal (base 16) representations as
\[ \text{Msg} \sim \text{KeyGen}(\text{RxM}, \text{RxL}) \]
\[ = \text{mdt1mdt2mdt3mdt4|mdt1+16mdt2+16mdt3+16mdt4+16|mlds1mlds2mlds3} \]
\[ = \text{mds4|mlds1+16mds2+16mds3+16mds4+16} \]
\[ = dv1dv2dv3dv4 \equiv RV \ldots \ldots \ldots \ldots (6) \]
Where,
dv1dv2dv3dv4 is the hexadecimal (base 16) notation.
As will be seen in (7), PW – KeyGen(RV, RxM) denotes the KeyGen output performed over PW using the previously generated RV and RxM to indicate a bit location in PW and concatenates these bits to form a 16-b Key. The resulting Key would then be expressed as,
\[ PW \sim \text{KeyGen}(\text{RV}, \text{RxM}) \]
\[ = \text{pdv1pdv2pdv3pdv4|pdv1+16pdv2+16pdv3+16pdv4+16|pdv1+16pdv2+16pdv3+16pdv4+16} \]
\[ = hw1hw2hw3hw4 \equiv \text{Key} \ldots \ldots \ldots \ldots (7) \]
where hw1hw2hw3hw4 is the hexadecimal (base 16) notation. The key generation requires two steps of (6) and (7), which is represented by Key = PW – Msg – KeyGen(Rx).

a). Encoding procedure of message:
Let the 32-b message (Msg) represent as
\[ \text{Msg} = \text{MsgM||MsgL} \ldots \ldots \ldots \ldots \ldots (8) \]
where MsgM and MsgL are 16 MSBs and 16 LSBs. By utilizing Key in (7), we perform the XOR operation for cover coding MsgM and MsgL, respectively. That is,
\[ \text{CCMsgM} = \text{MsgM} \oplus \text{PW} - \text{Msg} - \text{KeyGen(Rx)} \ldots \ldots \ldots \ldots \ldots (9) \]
\[ \text{CCMsgL} = \text{MsgL} \oplus \text{PW} - \text{Msg} - \text{KeyGen(Rx)} \ldots \ldots \ldots \ldots \ldots (10) \]
Where,
\( \oplus \) - logic EX-OR
Consequently, the cover-coded message is
\[ \text{CCMsg} = \text{CCMsgM} \text{CCMsgL} \ldots \ldots \ldots \ldots \ldots (11) \]
For convenience sake, the steps of (9)–(11) are briefly expressed as
\[ \text{CCMsg} = \text{Msg} \oplus \text{PW} - \text{Msg} - \text{KeyGen(Rx)} \ldots \ldots \ldots \ldots \ldots (12) \]
Finally, the original message can be decoded and recovered via the XOR operation as follows:
\[ \text{Msg} = \text{CCMsg} \oplus \text{PW} - \text{Msg} - \text{KeyGen(Rx)} \ldots \ldots \ldots \ldots \ldots (13) \]
The security is therefore significantly enhanced by this low complexity technique.

Security Analysis of the Key Generation Function:
To investigate the possibility of the key decryption, we consider the scenarios as follows.

**Scenario 1:**
RxM is hacked and modified as 0000h (base 16). In this scenario, we have
\[ RV = \text{Msg} - \text{Key Gen}(0000h, \text{RxL}) \]
\[ = m0m0m0m0|m16m16m16m16|mlds1mlds2mlds3 \]
\[ = mds4|mlds1+16mds2+16mds3+16mds4+16 \ldots \ldots \ldots \ldots (14) \]
It is observed that RV \( \in \{00XYh, 0FXYh, F0XYh, FFXYh\}. \) Each case has an equal probability of 1/4.

**Case 1:**
\[ RV = 00XYh: \) The generated key is computed as Key = PW – Key Gen(0000h, 0000h)
\[ = p0p0pxy||p16p16px+16py+16||p0p0p0p0||p16p16p16p16 \ldots \ldots \ldots \ldots (15) \]
We consider the following conditions of X and Y.
When X = 0 and Y = 0, the probability of key decryption is 1/16 \times 1/16 = 1/256. Note that the probability of X = 0 (Y = 0) is 1/16.
When X = 0 and Y \( \_\_ \_ = 0 \) (X \( \_\_\_ = 0 \) and Y = 0), the probability of key decryption is 1/16 \times 1/16 \times 1/24 = 15/256. The probability that the key is decoded in case 1 is
\[ \text{Pr(Case 1)} = 1/4 \times (1/210+ 2 \times 15/212 + 225/214) = 0.0055 \]

**Case 2:**
\[ RV = 0FXYh: \) Key = PW – Key Gen(0FXYh, 0000h) = p0p15pxy \[ m16p31px+16py+16||p0p0p0p0||p16p16p16p1 \ldots \ldots \ldots \ldots (16) \]
The conditions of X and Y are considered as follows.
When $X = 0$ and $Y = 0(X = F$ and $Y = F$), the probability of key decryption is $1/16 \times 1/16 \times 1/24 = 1/212$.

When $X = 0$ and $Y = F \ (X = F$ and $Y = 0)$, the probability of key decryption is $1/16 \times 1/16 \times 1/24 = 1/212$.

When $X = 0$ and $Y = 0 \ (X = F$ and $Y = 0, F)$, the probability of key decryption is $1/16 \times 14/16 \times 1/26 = 7/213$.

When $X = 0, F$ and $Y = 0 \ (X = 0, F$ and $Y = F)$, the probability of key decryption is $1/16 \times 14/16 \times 1/26 = 7/213$.

When $X = 0, F$ and $Y = F$, the probability of key decryption is $14/16 \times 1/24 = 7/144$.

The probability of key decryption in case 2 is $Pr(\text{Case } 2) = 1/4 \times 1/24 = 1/96$.

Case 3:

$RV = F0XYh$:  
$Key = PW - KeyGen(F0XYh, 0000h) = p15p0p0p0pY \ || p31p16pX + 16pY + 16[p0p0p0p0]p16p16p16p16$ (17)

As the discussion in case 2, we have the probability of key decryption as $Pr(\text{Case } 3) = 0.0019$.

Case 4:

$RV = FFXYh$:  
$Key = PW - KeyGen(FFXYh, 0000h) = p15p15p15pY[p31p16pX + 16pY + 16[p0p0p0p0]p16p16p16p16$  (18)

Similarly, the probability of key decryption in case 4 is $Pr(\text{Case } 4) = 0.0019$.

According to cases 1–4, we have the probability that the key is decoded in scenario 1 is $0.0055 + 3 \times 0.0019 = 0.0112$.

Scenario 2:

RxM and RxL are both hacked and modified as 0000h (base 16).

$RV = M16 - KeyGen(0000h, 0000h) = m0m0m0m0m16m16m16m16m0m0m0m0$ (19)

It implies that $RV \in \{0000h, 0F0Fh, F00Fh, FFFFh\}$. Each case has an equal probability of 1/4.

Case 1:

$RV = 0000h$:  
The generated key is computed as $Key = PW - KeyGen(0000h, 0000h) = p0p0p0p0p0p16p16p16p16p16p16p16p16p16p16p16p16p16p16p16p16p16p16p16$ (20)

The probability of key decryption in case 1 is $Pr(\text{Case } 1) = 1/4 \times 1/22 = 1/24$.

Case 2:

$RV = 0F0Fh$:  
The generated key is computed as $Key = PW - KeyGen(0F0Fh, 0000h) = p0p15p0p0p15[p16p16p16p16p0p0p0p0]p16p16p16$ (21)

The probability of key decryption in case 2 is $Pr(\text{Case } 2) = 1/4 \times 1/24 = 1/26$.

Case 3:

$RV = F00Fh$:  
The generated key is computed as $Key = PW - KeyGen(F00Fh, 0000h) = p15p15p15p15[p31p31p31p31p0p0p0p0]p16p16p16p16$ (22)

The probability of key decryption in case 3 is $Pr(\text{Case } 3) = 1/4 \times 1/24 = 1/26$.

Case 4:

$RV = FFFFh$:  
The generated key is computed as $Key = PW - KeyGen(FFFFh, 0000h) = p15p15p15p15[p31p31p31p31p0p0p0p0]p16p16p16p16$ (23)

The probability of key decryption in case 4 is $Pr(\text{Case } 4) = 1/4 \times 1/24 = 1/26$.

In summary, the probability that the key is successfully decoded in scenario 2 is $1/24 + 3 \times 1/26 = 0.1094$.

b) Hardware implementation

According to the mutual authentication protocol described in the previous section, the proposed hardware architecture for KeyGen functions is shown in Fig. 1. In this figure, the message $M$, password $PW$, and random number $Rx$ serve as the input of KeyGen operation via the 1_to_2 Mux modules. The output $Key_{xy}$ performs the XOR operation with the outputs of $Msgx$ and $PWx$ via the 1_to_2 Mux modules to generate $CCMsgM$, $CCMsgL$, $CCPWM$, and $CCPW$. Finally, $CMOutM$, $CMOutL$, $CPOutM$, and $CPOutL$ are generated by the CRC-16 module. Simulations of the proposed design were conducted in the Altera Quatrus II design environment and implemented in the Altera Cyclone II EP2C70F896C6 field-programmable gate array (FPGA) on an Altera DE Board. The simulation results of the KeyGen function are shown in Fig. 1. Assume that $Msg = 12345678$ (hex) and $PW = ABCDEFG01$ (hex). The random number is generated as $Rx = 5761AD5C$ (hex). In Fig. 1, the key is produced as 31F4 (hex) based on the KeyGen algorithm. They execute the XOR operations with $MsgM$ and $MsgL$ so as to generate $CCMsgM = 23C0$ (hex) and $CCMsgL = \ldots$
678C (hex), respectively. Finally, the CRC-16 codes are computed for CCMsgM and CCMsgL. When swt_1 turns high, the outputs are CCMsgM||CRC(CCMsgM) and CCMsgL||CRC(CCMsgL). Note that swt_1 is the control signal. Therefore, CMOutM = 23C04883 (hex), and CMOutL = 678CD12D (hex). As shown in Fig. 1, PWM and PWL perform XOR operations to generate CCPWM = 9A39 (hex) and CCPWL = DEF5 (hex). The CRC-16 codes for CCPWM and CCPWL are then computed. When swt_1 turns high, the outputs are CCPWM||CRC(CCPWM) and CCPWL||CRC(CCPWL). Accordingly, CPOutM = 9A395C9C (hex), and CPOutL = DEF54631 (hex).

The verified Verilog code was downloaded on an Altera Cyclone II FPGA in the Altera DE2 board. This Altera DE2 board included an Altera Cyclone II FPGA and various onboard components. The FPGA implementation and verification platform are shown in Fig. 2. Again, let random number Rx be 5761AD5C (hex). Based on the KeyGen algorithm, we yield CCMsgM||CRCMsgM = 23C04883 (hex), CCMsgL||CRCMsgL = 678CD12D (hex), CCPWM||CRCPWM = 9A395C9C (hex), and CCPWL||CRCPWl = DEF54631 (hex).

![Fig. 1. Simulation results of the KeyGen function](image)

**IV. RESULT AND CONCLUSION**

From this we observed that difference between the existing and proposed method is, golay encoder replaces the CRC in the receiver side. In CRC only bits are checked and fed to Receiver but in golay coding which provides additional security for data(message) and also the instant correction mechanism for error control. Also it provides better time to compute the authentication and encryption for entire cycle until it complete its operation.

![Fig. 2. Result displayed by Altera DE2-70 FPGA board.](image)

![Fig. 3. Simulation output with lesser time cycle](image)

**V. REFERENCES**


