ISSN:0975-9646



# Implementation of a Unique Light Weight Time Stamp Based Security Protocol for Wireless Adhoc Network

[1]Rajeev Dhawan , [2]Swati Singhal

[1]*Student, Computer Science and Engineering Deptt, PDM College of Engineering, Bahadurgarh, Haryana (India)*
[2]*Assistant Professor, PDM College of Engineering, Bahadurgarh, Haryana (India)*

**Abstract-In recent years after the Development and Establishment of MANET [1] according to the Industry Standards and based on optimal performance. We have optimally developed and designed these ad-hoc networks which may be infrastructured or infrastructureless network and the devices in the network communicated using IEEE 802.1x MAC standard [4].**

## 1. INTRODUCTION

We have already discussed authentication protocols in past like Das Authentication protocol and Nyang Authentication protocol [2, 3, 5]. The Timestamp based Security protocol for wireless adhoc network is somewhat different from these two Das and Nyang Authentication protocol. The cryptography is Oblate public key cryptography. The Detailed discussion of Das and Nyang Authentication protocol is there in this paper.

## 2. REVIEW OF THE DAS AND NYANG AUTHENTICATION PROTOCOLS

### 2.1. The Das Authentication Protocol

The Das two-factor user authentication protocol [4] is based on smart card technologies and uses a hash function for providing security services. The proposed system assumes that an intruder cannot extract data from a compromised smart card or sensor node. The smart card uses the data stored upon it for on-card computation in order to protect the system against impersonated gateway node attacks. The protocol consists of three phases: registration, login, and authentication.

**(1) Registration Phase**

The gateway node generates two system secret parameters: $K$ and $x_a$ for registration use. The system secret $x_a$ is stored on the sensor nodes before their deployment and distribution into the WSN. A user $U_i$, who wants to register with the WSN, submits his identity $ID_i$ and password $PW_i$ to the gateway node in a secure way. The gateway node computes an additional secret that binds the user identity and password to the system secret $K$: $N_i = h(ID_i||PW_i) \oplus h(K)$. The hash function, $h()$, and the parameters: $\{ID_i, N_i, h(PW_i), x_a\}$ are stored onto a new smart card that is given to the user. The user has no knowledge of the secret parameters: $N_i$, $K$, and $x_a$ on the card.

**(2) Login Phase**

To log in and access data from the WSN, the user $U_i$ inserts his smart card into a terminal and keys $ID_i$ and $PW_i$. The smart card validates the entered $ID_i$ and $PW_i$ by comparing them with the stored ones. If the verification is successful, the smart card performs the following operations.

1. Compute $DID_i = h(ID_i||PW_i) \oplus h(x_a||T)$, where $T$ is the current timestamp of $U_i$'s terminal;
2. Compute $C_i = h(N_i||x_a||T)$;
3. Send $<DID_i, C_i, T>$ to the gateway node.

**(3) Verification Phase**

Upon receiving the login request message $<DID_i, C_i, T>$ at time $T^*$, the gateway node authenticates $U_i$ with the following procedures.

1. Validate $T$. The gateway node proceeds to next step if $T^*-T \leq \Delta T$, where $\Delta T$ is the expected time interval for transmission delay. Otherwise, it rejects the request as stale;
2. Compute $h(ID_i||PW_i)^* = DID_i \oplus h(x_a||T)$ and $C_i^* = h((h(ID_i||PW_i)^* \oplus h(K))||x_a||T)$;
3. Validate $C_i$. The gateway node accepts the login request if $C_i = C_i^*$. Otherwise, it rejects the request;
4. Send $<DID_i, A_i, T'>$ to the nearest sensor node (e.g., $S_n$) to respond to $U_i$'s query, where $T'$ is the current timestamp of the gateway node's system, and $A_i = h(DID_i||S_n||x_a||T')$;
5. $S_n$ first verifies $T'$ as in Step 1 above. It computes $A_i^* = h(DID_i||S_n||x_a||T')$ and validates whether $A_i = A_i^*$. If the verification is successful, $S_n$ sends the requested data to $U_i$.

### 2.2. Analysis of the Das Authentication Protocol

From the description of the protocol, we can see that there is no key established between the user and gateway node or sensor node and therefore no secure channel to protect the requested data. This is, perhaps, because the protocol was not intended for use in adversarial environments.

In situations where there are strong adversaries to contend with, there are several weaknesses of the Das protocol that we can identify. Foremost among them is the distribution of the system secret $x_a$ onto every smart card and sensor node [7] in the system. Das explicitly assumes that extraction of $x_a$ from the smart card is difficult. While this may be true, it should be considered a possibility, as should the risk of $x_a$ being extracted from a captured or compromised sensor node. Even though the protocol uses two system secrets, learning one of them, $x_a$, is sufficient to break the security.

If a single node or smart card in the system is captured and compromised, an adversary can use the extracted secret $x_a$ to create a fake query. It would do so by creating its own $A_i = h(DID_i||S_n||x_a||T)$ and sending a forged $<DID_i, A_i, T>$ to the sensor node $S_n$. Since the sensor node only verifies $A_i$ and $T$, and does not verify $DID_i$, the adversary can use an old $DID_i$ or simply put anything in the $DID_i$ field to generate a new $A_i$ with $\{T, S_n, x_a\}$ where $T$ is the current timestamp, $S_n$ is the sensor node's identity, and $x_a$ is the compromised system secret. That is, the verification at the sensor node is only verifying that the query was generated

by someone who knows the hash function *h()* and the secret $x_a$. All other parameters are either public, sent in the clear, or, in the case of $DID_i$, superfluous. As a result, adversaries can forge $<DID_i, A_i, T>$ and request data from any sensor node [7]. This makes the system weak with respect to robustness, tolerance, and resilience.

There is also a weakness with respect to an insider attack, for which the attacker does not need to be able to extract system secrets from a node or smart card. In [4], it is claimed that an attacker cannot generate a $DID_i$ with a new timestamp without knowing the user's password $PW_i$ and the system secret $x_a$. We show that this is not the case. First, the inside attacker (e.g., $U_j$) eavesdrops the request message $<DID_i, C_i, T>$ sent by the user $U_i$. The attacker resets his system time to $T$ and uses his smart card to generate a new message $<DID_j, C_j, T>$. Next, he computes $h(x_a||T)$ by $h(x_a||T)=DID_j \oplus h(ID_j||PW_j)$ using his own identity $ID_j$ and password $PW_j$, and retrieves the user $U_i$'s $h(ID_i||PW_i)$ by $h(ID_i||PW_i) = DID_i \oplus h(x_a||T)$. The attacker now has the hash of $U_i$'s identity and password, a constant that is intended to provide security by binding a user's identity to queries. Finally, the attacker can generate a new request message $<DID_j', C_j', T'>$ using his own smart card and current timestamp, and further get a new $h(x_a||T')$ and $DID_i'$ with the current timestamp $T'$ by $DID_i'=h(ID_i||PW_i) \oplus h(x_a||T')$. By these means, the attacker can generate a valid $DID_i$ for any time. Now, without being able to calculate a corresponding, valid $C_i$, which depends on the system secret $K$, the inside attacker cannot forge a complete query $<DID_i, C_i, T>$ for any $T$, but clearly this represents an unintended design flaw in the protocol that weakens the system. In the case where the smart card itself is attacked, the adversary can similarly use $h(ID_i||PW_i)$ along with an extracted $N_i$ to recover $h(K)$. This could expose $K$ to further attack if there were any gain in doing so.

Other shortcomings of the protocol in an adversarial environment include the unilateral authentication, unprotected data transfer, and fixed password. In the protocol, the gateway node authenticates the user but the gateway itself is not authenticated. Without authenticating the gateway, an intruder can impersonate the gateway to send falsified data to the user. Also, since the data is not bound to the identity of the sensor node or the requestor it is easy for an adversary to modify or misdirect the data. For example, sensor data could be sent to a different user than original requestor. In addition, users cannot freely change passwords since $N_i$ is pre-computed and stored into the smart card for on-card computation. More detailed research on smart card-based remote user authentication can be found in [6].

## 3. THE NYANG AUTHENTICATION PROTOCOL

The Nyang authentication protocol [7] is an augmentation of the Das protocol. It uses $h(ID_i||PW_i||x_a)$ instead of $h(ID_i||PW_i)$ for the system secret $N_s$ calculation in order to prevent an off-line password guessing attack in the Das protocol (see [7]) and introduces a symmetric key $x_n$ for the senor node $S_n$ in order to establish a secure channel between the user and sensor node.

Although the Nyang protocol solves some issues in the Das protocol, it still suffers sensor node and smart card compromising attacks since its system secret $x_a$ is stored on both of sensor node and smart card, which makes the system weak with respect to tolerance and resilience. In addition, the Nyang protocol still keeps many features as same as the Das protocol such as unilateral authentication, fixed password, smart card and sensor node must be equipped with a tamper-proof module and on-card computation, etc.

In order to improve the Das and Nyang protocols and provide strong authentication under the WSN resource-constrained environment, we present the following strong user authentication protocol and lightweight user authentication protocol.

## 4. OUR APPROACH LIGHTWEIGHT USER AUTHENTICATION PROTOCOL

The lightweight user authentication protocol we propose is based on smart card technology. The smart card can be integrated into the user's mobile devices with self-lock or destroy functionality for physical security. The protocol uses a hash to protect the system secret and consists of three phases: registration, authentication, and secure data transfer phases. Figure 3 depicts the data flow of the lightweight user authentication protocol.

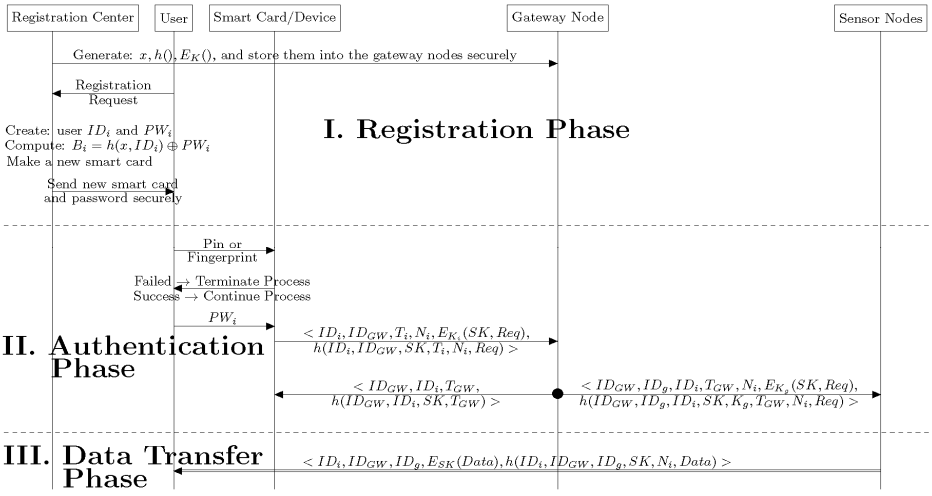


**Figure 3: Data Flow of the Lightweight User Authentication Protocol**

### 4.1 Registration Phase

The system generates the following parameters: $x$, $h()$, and $E_K()$, and stores them into the gateway nodes, where $x$ is the system secret. The registration center creates a user identity $ID_i$ and remote access password $PW_i$, computes a long-term key $K_i$ as $K_i=h(x, ID_i)$ and $B_i=K_i \oplus PW_i$ for user $U_i$, stores $ID_i$, $B_i$, $h()$, and $E_K()$ onto a new smart card, and sends the smart card with the password to the user in a secure way. The user knows nothing about the system secret $x$ and $K_i$ if he does not compromise his smart card physically and extract the parameter $B_i$ but can change passwords after receiving the smart card.

For password change, the smart card needs to update the data $B_i$ stored in the card by $B_i'=B_i \oplus PW_i \oplus PW_i'$, where $PW_i$ is the old password and $PW_i'$ is the new password. To do this, the protocol has two options. The first option does not require connecting to the registration centre. In it, the user needs to connect to the registration centre to re-setup his password if he inputs an incorrect one. The second option is that the user needs to go through the following authentication procedure and let the gateway node verify his old password first.

### 4.2 Authentication Phase

The authentication phase contains the following three steps.

**Step 1:** The user $U_i$ inserts his smart card into his mobile device and keys a pin or scans his finger for smart card access authentication. If the pin or fingerprint verification is successful, the user then keys the remote access password. Unlike the strong user authentication, the smart card authenticates the user with the pin or fingerprint but not the remote access password for smart card access. This can provide another security feature such as stealing and compromising the smart card since we do not directly store the remote access password and system secret in the card.

The smart card recovers the user's long term key $K_i$ by $K_i=B_i \oplus PW_i$, and sends the following request message to the gateway node $GW$:

***Message 1.*** $U_i \rightarrow GW$:

$<ID_i, ID_{GW}, T_i, N_i, E_{Ki}(SK, Req), h(ID_i, ID_{GW}, SK, T_i, N_i, Req)>$, where $T_i$ is the current timestamp of $U_i$'s device, $SK$ is a session key generated by the smart card.

**Step 2:** Upon receiving the request message at time $T_{GW}$*, the gateway node $GW$ validates the destination identity $ID_{GW}$ and the timestamp $T_i$ by comparing $T_{GW}$*-$T_i \leq \Delta T$. If the verification is successful, $GW$ recovers the user's long-term key $K_i$ by $K_i=h(x, ID_i)$ and decrypts the ciphertext with $K_i$. It then validates the user and received message by checking the hash value. If they are correct, it sends the following authentication message back to the smart card:

***Message 2.*** $GW \rightarrow U_i$ :

$<ID_{GW}, ID_i, T_{GW}, h(ID_{GW}, ID_i, SK, T_{GW}, Req)>$, where $T_{GW}$ is the current timestamp of the gateway node. Meanwhile, the gateway node sends the following message back to the targeted sensor nodes:

***Message 3.*** $GW \rightarrow S_g$ :

$<ID_{GW}, ID_g, ID_i, T_{GW}, N_i, E_{Kg}(SK, Req), h(ID_{GW}, ID_g, ID_i, SK, K_g, T_{GW}, N_i, Req)>$.

where $K_g$ is the group key of the sensor nodes having the requested data, $ID_g$ is their group identity, $T_{GW}$ is the current timestamp of the gateway node. $K_g$ is managed by the gateway and could be updated periodically depending on the different applications. $K_g$ could be also a shared key between the gateway node and a specific sensor node under the situation when the requested data only is stored on that sensor node. For key management among sensor nodes and GWs such as group key and shared key establishment, we can use existing technologies (e.g., IKDM [7] and Key Evolution [ 2, 3, 7]), which is beyond the scope of this paper.

**Step 3:** After receiving the authentication message $<ID_{GW}, ID_i, T_{GW}, h(ID_{GW}, ID_i, SK, T_{GW}, Req)>$ at time $T_i$*, the smart card validates the destination identity $ID_i$, the timestamp $T_{GW}$ by comparing $T_i$*-$T_{GW} \leq \Delta T$, and the received message by checking the hash value. If the authentication is successful, it waits for the data from the sensor nodes.

### 4.3 Secure Data Transfer Phase

Upon receiving the message sent by the gateway node at time $T$*, the sensor nodes check the group identity $ID_g$ and validate the timestamp $T_{GW}$ by $T$*-$T_{GW} \leq \Delta T$. The sensor nodes decrypt the ciphertext with the group key $K_g$ and validate the received message by checking the hash value. If the authentication is successful, the sensor nodes send the requested data to the user through the following secure channel:

***Message 4.*** $S_g \rightarrow U_i$ :

$< ID_g, ID_i, ID_{GW}, E_{SK}(Data), h(ID_g, ID_i, ID_{GW}, SK, N_i, Data)>$.

The user decrypts the data with the session key $SK$ after receiving the message from the sensor nodes and validates the received message by checking the hash value.

## 5. ANALYSIS OF THE PROTOCOLS

In this section, we analyze the security of the new authentication protocol and demonstrate its strength with respect to security and efficiency.

### 5.1 Security Characteristics

Some security characteristics of the new protocol are examined below.

#### 5.1.1 Security of the System Secret

The security of the new authentication protocol relies on the system secret $x$. In the proposed protocol, only the gateway nodes contain the system secret $x$ which is protected with tamper-resistant technology. It is difficult for an intruder to re-compute or recover a user $U_i$'s long-term key $K_i$ based only on the identity $ID_i$ without knowing $x$. In order to recover the system secret $x$, the user needs break the hash function from $K_i=h(x, ID_i)$.

We stress that the proposed protocol do not store any system secret on the smart cards or sensor nodes. The intruders cannot get any information about the system secret by capturing and compromising the sensor nodes since the Shannon mutual information $I(x, K_g)=0$ and $I(x, M)=0$, where $K_g$ is the group key of the sensor nodes, and $M$ is the authentication message sent to the sensor nodes [7]. In addition, since the group key is updated periodically, this means captured and compromised nodes can only impact system security for applications related to the group key during a limited period. This provides forward and backward security for the system and provides for a system that has strong tolerance, robustness, and resilience.

### 5.1.2 Secure Channel

The new authentication protocol creates a secure channel between the user and gateway node or sensor nodes during authentication by establishing a shared session key between them. This provides protection for transferring data from the sensor nodes to the user, securing it against attacks such as eavesdropping, message modification, and data misdirection.

**Hardware & Software to Be Used**

**Hardware Devices**

- Wireless Router
- Wireless card

**Software**

- Intel Proset/Wireless software
- Cisco Acu-client

## CONCLUSION:

We have already discussed MANET [1] and about its simulation in this Research paper. We have discussed and Implemented Security protocol that is designed for wireless adhoc networks [7]. It is a light weight protocol and is a time stamp based security protocol. The broadcast message and the nodes have already been discussed [7].

## REFERENCES:

[1] F. Baker, "An outsider's view of MANET," Internet Engineering Task Force document, 17 March 2002.

[2] C. Barrett et al., "Characterizing the Interaction Between Routing and MAC Protocols in Ad-hoc Networks," *Proc. MobiHoc 2002*.

[3] J. Broch et al., "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," *Proc. Mobicom '98*.

[4] M. S. Corson et al., "Internet-Based Mobile Ad Hoc Networking," *IEEE Internet Computing*, July-August 1999.

[5] L. M. Feeney, "A Taxonomy for Routing Protocols in Mobile Ad Hoc Networks," Swedish Institute of Computer Science Technical Report T99/07, October 1999.

[6] S. Kurkowski, et al., "MANET Simulation Studies: The Incredibles," *ACM SIGMOBILE Mobile Computing and Communication Review*, Vol. 9, Issue 4 (October 2005).

[7] C. E. Perkins, *Ad Hoc Networking*. New York: Addison-Wesley, 2001.