JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 30 No. 12, pp. 205-213, December 2025
https://doi.org/10.9708/jksci.2025.30.12.205

A Secure and Efficient User Authentication Protocol for IoT Environments

Mi-Og Park*

*Professor, Dept. of Computer Engineering, Sungkyul University, Anyang, Korea

[Abstract]

In this paper, the security limitations and design flaws of the Li protocol proposed in 2023 are analyzed, and a new authentication protocol is proposed that addresses these concerns is proposed. The Li protocol analyzed in this paper contains a structural flaw that allows all legitimate users to directly access the server's secret key, which compromises the integrity of authentication and the confidentiality of the entire system, posing a critical security threat. Furthermore, the protocol is vulnerable to attacks such as smart-card lost attack, forward secrecy, session key exposure, ephemeral secret leakage attack, and key impersonation attack. This paper improves upon these issues and proposes a lightweight authentication protocol suitable for IoT environments. The proposed protocol, through security and performance analysis compared with related protocols, satisfies high security attributes while minimizing both computational and communication costs. Therefore this study is well-suited as a secure and practical IoT security authentication protocol.

▸ **Key words**: Server key exposure, Forward secrecy, Smart-card lost attack, Ephemeral secret leakage attack, IoT environments

[요 약]

본 논문에서는 2023년의 Li의 프로토콜에 대한 보안적 한계와 설계상의 문제를 분석하고, 이를 개선한 새로운 인증 프로토콜을 제안한다. 본 논문에서 분석한 Li의 프로토콜은 정당한 모든 사용자가 서버의 비밀키를 직접적으로 알 수 있는 구조적 결함으로 인하여 인증 무결성과 시스템 전체의 기밀성을 훼손하는 치명적인 보안 위협이 존재한다. 또한, 스마트카드 분실 공격, 전방향 안전성, 세션키 노출, 임시 비밀정보 유출 공격, 키 위장 공격 등에 취약하다. 본 논문에서는 이러한 문제를 개선하고, IoT 환경에 적합한 경량의 인증 프로토콜을 제안한다. 제안 프로토콜은 기존 프로토콜들과의 안전성과 성능 분석을 통해, 높은 수준의 보안 속성, 계산 비용과 통신 비용의 최소화를 만족한다. 그러므로 본 연구는 안전하고 실용적인 IoT 보안 인증 프로토콜로 적합하다.

▸ **주제어**: 서버의 비밀키 노출, 전방향 안전성, 스마트카드 분실 공격, 임시비밀정보 유출 공격, IoT 환경

• First Author: Mi-Og Park, Corresponding Author: Mi-Og Park
*Mi-Og Park (mopark777@daum.net), Dept. of Computer Engineering, Sungkyul University
• Received: 2025. 10. 02, Revised: 2025. 11. 21, Accepted: 2025. 12. 11.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

사물인터넷(Internet of Things, IoT)의 급속한 발전은 수많은 장치들이 상호 연결되는 환경을 조성하고 있으며, 이러한 환경에서 안전한 사용자 인증은 보안의 핵심 요소로 간주된다. 그러나 IoT 환경은 일반적인 IT 시스템과 달리 제한된 연산 능력, 낮은 전력 자원, 불안정한 네트워크, 물리적 노출 위험 등 다양한 제약 조건을 수반하기 때문에, 기존의 전통적인 인증 방식만으로는 충분한 보안성을 확보하기 어렵다. 이러한 배경에서 최근에는 패스워드, 스마트카드, 생체정보를 결합한 3요소 인증(Three-Factor Authentication, 3FA) 방식에 대한 연구가 활발히 진행되고 있다. 이러한 배경 속에서 2017년 Dhillon과 Kalra[1]는 IoT 환경에 적합한 경량 인증 프로토콜을 제안하였으며 자신들의 프로토콜은 알려진 다양한 공격에 대한 저항성을 가진다고 주장하였다.

그러나 2020년 Lee 등 프로토콜[2]은 Dhillon과 Kalra의 프로토콜이 모바일 기기 탈취 공격에 취약하고, 사용자 가장(user impersonation) 공격을 막지 못하며, 인증서 폐기(revocation) 메커니즘의 부재 문제를 지적하였다. 이에 Lee 등 프로토콜은 3FA에 기반한 익명성 인증 프로토콜을 제안하였고, 자신들의 프로토콜은 알려진 다수의 공격에 대응한다고 주장하였다. 하지만 2023년의 Li 등의 프로토콜[3]은 Lee 등의 프로토콜이 중간자 공격(man-in-the-middle attack)과 사용자 가장 공격에 대한 방어가 미흡하며, 세션 키 보호 및 전방향 안전성의 문제가 있다고 지적하였다. Li 등의 프로토콜은 이러한 문제를 해결하기 위해 프로토콜 구조를 개선하여 보안성과 효율성을 동시에 향상시키는 방안을 제시하였다.

한편 2021년의 Chang 등의 프로토콜[4]에서도 Lee 등의 프로토콜의 구조적 한계를 분석하고, 해당 프로토콜이 여전히 사용자 익명성과 추적 불가능성(un-traceability)을 완전히 보장하지 못하고, 사용자 역추적이 가능한 구조적 취약점을 내포하고 있다고 밝혔다. 또한, 계산 비용의 불균형으로 인해 IoT 단말에 과도한 연산 부담이 발생한다는 점도 함께 지적하였다. Chang 등의 프로토콜은 이러한 문제를 해결하기 위하여 임시 키 기반 DH 키 교환구조, 블라인드 서명, 생체정보 보호 매커니즘을 강화함으로써 사용자 프라이버시 보호, 경량성, 상호인증을 동시에 달성하는 새로운 프로토콜을 제안하였다. 그러나 2025년의 Lo 등의 프로토콜[5]은 Chang 등의 프로토콜에서도 여전히 추적 불가능성과 전방향 안전성의 문제가 있다고 지적하였다. 2024년에 Arpitha 등의 프로토콜[6]은 IoT 환경에서의 익명성 및 강력한 인증 프로토콜을 제안하였다. 이 인증 프로토콜에서는 IoT 건강 관리 애플리케이션에서 사용자 프라이버시 보호와 보안성을 강화하는 방안을 다루고 있으며, Lee 등 프로토콜의 보안 취약점들에 대하여 간략히 언급하였다.

2021년에 Sahoo 등의 프로토콜[7]도 IoT 환경의 헬스케어 시스템의 보안 취약점을 해결하기 위한 인증 시스템을 제안하였고, AVISPA 도구를 사용한 시뮬레이션 결과 해당 프로토콜은 안전하다고 주장하였다. 그러나 2022년 블록체인 기반의 IoT 인증 프로토콜을 제안한 Mirsaraei 등의 프로토콜[8]은 Sahoo 등의 프로토콜이 내부자 공격, 서버 가장 공격, 위조 공격에 취약하며, IoT 장치의 자원 제약을 고려하지 못한 비효율성 및 높은 계산 비용 등의 문제를 안고 있다고 지적하였다. 2023년에 Li의 프로토콜[9]은 Mirsaraei 등의 프로토콜 역시 추적 불가능성, 전방향 안전성, 키 위장 공격, 임시 비밀정보 유출 공격(ESL)의 취약하다는 점을 제기하였다. 이에 따라 Li의 프로토콜은 이러한 문제를 해결하기 위한 새로운 프로토콜을 설계하고, 패스워드 추측 공격과 세션 키 보호를 강화하며, 동시에 사용자 익명성 및 시스템 효율성 향상을 도모하는 방안을 제안하였다. 본 논문에서도 Li의 프로토콜을 분석한 결과, Li의 프로토콜은 설계상의 결함으로 인해 정당한 모든 사용자에게 서버의 비밀키가 노출되는 심각한 보안 취약점이 존재함을 확인하였다. 이로 인해 해당 프로토콜은 서버 가장 공격, 전방향 안전성, 임시 비밀정보 유출 공격 등 알려진 여러 공격에 안전하지 않다.

본 논문에서는 이러한 Li의 프로토콜[9]의 문제를 해결하고 개선하기 위하여, 2장에서 Li의 프로토콜의 각 단계를 살펴보고, 3장에서 해당 프로토콜의 문제를 분석한다. 4장에서는 문제 해결을 위한 개선된 경량 프로토콜을 제안하고, 5장에서 제안 프로토콜의 안전성과 계산 비용, 전송 비용 등을 비교·분석한다. 마지막으로 제6장에서는 제안 프로토콜이 보안성과 경량성을 동시에 만족하며, IoT 환경에 적합함을 종합적으로 논의한 후, 결론을 제시한다.

II. Li's Authentication Protocol

Li의 프로토콜[9]의 구성은 등록 단계, 로그인과 인증 단계, 그리고 패스워드 변경단계로 이루어진다. 초기 단계의 서버는 타원 곡선상의 점 P를 기준점(base point)으로 사용하며, 장기 비밀키 X_s를 보유한다.

2.1 User registration phase

1.사용자 U_i가 자신의 ID_i와 패스워드 PW_i, 그리고 자신의 생체정보 b_i를 입력한다. 스마트카드는 $Gen(b_i) = (\sigma_i, \tau_i)$과 $PWD_i = h(ID_i \| PW_i \| \sigma_i)$을 계산하여 서버에 $\{ID_i, PWD_i\}$을 안전하게 전송한다.

2.서버는 난수 N_i를 생성하고 $Z_i = h(SID_j \| X_s \| ID_i) \oplus PWD_i, ERD_i = ID_i \oplus h(N_i \| SID_j), B_1 = N_i \cdot P, B_2 = X_s \oplus h(PWD_i \| ID_i), Au_i = h(SID_j \| ID_i) \oplus token$을 계산하여, $h(ERD_i)$와 N_i를 저장한 후, 사용자에게 $\{Z_i, ERD_i, B_1, B_2, Au_i\}$가 저장된 스마트카드 SC_i를 안전하게 전달한다.

3.사용자 U_i는 $f_i = h(PWD_i \oplus Z_i)$와 $E_i = Z_i \oplus h(PWD_i \| \sigma_i)$을 계산한 후, E_i와 f_i를 스마트카드에 추가로 저장하고, Z_i는 삭제한다. 스마트카드에는 최종적으로 $\{E_i, f_i, ERD_i, B_1, B_2, Au_i\}$가 저장된다.

2.2 Login and authentication phase

1.사용자 U_i가 자신의 ID_i와 패스워드 PW_i, 그리고 자신의 생체정보 b_i를 입력하여 $\sigma_i' = Rep(b_i, \tau_i)$, $PWD_i' = h(ID_i \| PW_i \| \sigma_i')$, $Z_i' = E_i \oplus h(PWD_i' \| \sigma_i')$, $f_i' = h(PWD_i' \oplus Z_i')$을 계산한 후 f_i'과 스마트카드의 f_i가 같은지 비교한다. 만약 두 값이 다르면 세션을 종료하고, 동일하면 난수 N_u와 타임스탬프 T_1을 생성하여 다음 값들을 계산한 후 $\{T_1, GID_i, M_1, B_3, B_4\}$를 서버에 전송한다.

$$X_s = B_2 \oplus h(PWD_i \| ID_i)$$
$$B_3 = ERD_i \oplus h(T_1 \| X_s)$$
$$G_i = Z_i \oplus PWD_i, \ V_1 = N_u \cdot B_1$$
$$B_4 = Au_i \oplus h(V_1 \| ID_i)$$
$$GID_i = V_1 \oplus h(ID_i \| T_1 \| X_s)$$
$$M_1 = h(G_i \| T_1 \| V_1)$$

2.메시지를 받은 IoT 서버는 타임스탬프 T_1의 타당성을 확인하여 타당하지 않으면 세션을 종료하고, 타당할 경우 $h(ERD_i')$을 계산한 후 저장된 $h(ERD_i)$와의 동일성 여부를 비교한다. 만약 동일하지 않으면 세션을 종료하고, 그렇지 않을 경우에는 $h(ERD_i)$에 매핑되는 난수 N_i을 검색하여 다음 값들을 계산한다.

$$ERD_i' = B_3 \oplus h(T_1 \| X_s)$$
$$ID_i' = ERD_i' \oplus h(N_i \| SID_j)$$
$$G_i = h(SID_j \| X_s \| ID_i)$$
$$V_2 = GID_i \oplus h(ID_i \| T_1 \| X_s)$$
$$Au_i' = B_4 \oplus h(V_2 \| ID_i)$$
$$token' = Au_i' \oplus h(SID_j \| ID_i)$$
$$M_1' = h(G_i \| T_1 \| V_2)$$

만약 계산한 M_1'과 전송받은 M_1이 동일하지 않으면 여기서 세션을 종료한다. 그렇지 않을 경우에는 난수 N_s와 타임스탬프 T_3을 생성하여 세션 키 SK_j를 계산한 후, $\{T_3, M_2, M_3\}$을 사용자에게 전송한다.

$$SK_j = h(V_2 \| N_s \| ID_i \| SID_j)$$
$$M_2 = SK_j \oplus h(X_s \| V_2 \| ID_i)$$
$$M_3 = h(G_i \| T_3 \| SK_j)$$

3.사용자는 타임스탬프 T_3의 타당성을 확인하여 타당하지 않으면 세션을 종료하고, 그렇지 않을 경우에는 $SK_i = M_2 \oplus h(X_s \| V_1 \| ID_i)$와 $M_3' = h(G_i \| T_3 \| SK_i)$을 계산하여 전송받은 M_3과 M_3'의 동일성 여부를 비교한다. 만약 같은 값이면 세션 키를 정당한 키로 인증하고, 그렇지 않을 경우에는 세션을 종료한다.

2.3 Password change phase

1.사용자 U_i가 자신의 식별자 ID_i와 패스워드 PW_i, 생체정보 b_i를 입력하면, 스마트카드는 σ_i', PWD_i', Z_i', f_i'을 계산하여, 계산한 f_i'과 f_i의 동일성 여부를 비교한다. 만약 두 값이 동일하지 않으면 세션을 종료하고, 그렇지 않을 경우에는 다음 계산을 진행한다.

$$\sigma_i' = Rep(b_i, \tau_i)$$
$$PWD_i' = h(ID_i \| PW_i \| \sigma_i')$$

Table 1. Notation

Symbol	Description
X_s	long term secret key of S_j
ID_i	identity of user i
PW_i	password of user i
b_i	biometrics of user i
$Gen(), Rep()$	key generation/recovery process of fuzzy extractor
S_j	J_{th} IoT server node
SID_j	identity of S_j
Au_i	authorization(access level) of user i
$h(.)$	one-way hash function
$\oplus$, $\|$	XOR and concatenation operation

$Z_i' = E_i \oplus h(PWD_i' \| \sigma_i')$

$f_i' = h(PWD_i' \oplus Z_i')$

2.사용자는 새로운 패스워드 PW_i^n과 생체정보 b_i^n을 입력한다.

3.스마트카드는 다음 값들을 계산한 후, 새롭게 계산한 E_i^n, f_i^n, τ_i^n, B_2^n을 각각 저장한다.

$Gen(b_i^n) = (\sigma_i^n, \tau_i^n)$

$PWD_i^n = h(ID_i \| PW_i^n \| \sigma_i^n)$

$Z_i^n = Z_i \oplus PWD_i' \oplus PWD_i^n$

$f_i^n = h(PWD_i^n \oplus Z_i^n)$

$E_i^n = Z_i^n \oplus h(PWD_i^n \| \sigma_i^n)$

$B_2^n = B_2 \oplus h(PWD_i' \| ID_i) \oplus h(PWD_i^n \| ID_i)$

III. Analysis of Li's Protocol

본 논문에서 분석한 Li의 프로토콜의 가장 심각한 문제는 로그인 단계에서 서버의 비밀키가 정당한 모든 사용자들에게 노출된다는 점이다. 이러한 문제로 인하여 Li의 프로토콜은 인증 프로토콜의 기본적인 설계뿐만 아니라 알려진 여러 공격에 대한 안전성 문제가 존재한다.

1. Security analysis

3.1.1 Server key exposure

Li의 프로토콜은 로그인 단계에서 카드 소유자를 확인한 후에 식 $X_s = B_2 \oplus h(PWD_i \| ID_i)$를 곧바로 계산한다. X_s는 서버의 비밀키이므로, Li의 프로토콜은 로그인 단계에서 서버의 비밀키 X_s가 정당한 모든 사용자들에게 노출되는 매우 심각한 문제가 존재한다. 정당한 사용자로 가장하여 등록한 공격자는 서버의 비밀키를 쉽게 알 수 있다. 서버의 비밀키 X_s를 알고 있는 공격자나 정당한 사용자는 서버의 생성 난수 N_i을 제외하고, 인증 단계의 모든 값을 계산할 수 있다. 또한, 세션 키 계산과정에서 서버의 생성 난수 N_s는 공격자가 획득하지 못한 값이라 할지라도 공격자 자신이 직접 생성한 난수 N_a를 이용하여 세션 키 $SK_a = h(V_2 \| N_a \| ID_i \| SID_j)$를 계산함으로써 정당한 서버로 가장할 수 있다. 추측한 ID_i' 값의 정확성은 $M_1' = h(G_i \| T_1 \| V_2)$식을 이용하여 검증할 수 있다.

3.1.2 Perfect forward secrecy

앞의 서버 키 노출 공격 절에서 분석한 바와 같이 Li의 프로토콜은 서버의 비밀키가 모든 사용자들에게 노출되는 문제가 존재한다. 그러므로 공격자는 서버의 생성 난수 N_i와 N_s를 모른다 할지라도 SK_j와 SK_i는 동일한 값이므로, 전송 메시지 M_2, GID_i, T_1, 그리고 서버의 비밀키 X_s와 추측한 ID_i'을 이용하여 세션 키를 다음과 같이 계산할 수 있다. SK_s 값의 정확성은 식 $M_3 = h(G_i' \| T_2 \| SK_i')$을 이용하여 검증 가능하며, 이때 G_i'은 서버의 계산식 $G_i' = h(SID_j \| X_s \| ID_i')$을 이용한다.

User U_i		*Server* S_j
Inputs ID_i, PW_i, b_i		Checks $(T_2 - T_1) \le \triangle T$
Computes $\sigma_i' = Rep(b_i, \tau_i)$		Computes $ERD_i' = B_3 \oplus h(T_1 \| X_s)$
$PWD_i' = h(ID_i \| PW_i \| \sigma_i')$		Computes $h(ERD_i')$ and retrieves N_i
$Z_i' = E_i \oplus h(PWD_i' \| \sigma_i')$		Computes $ID_i' = ERD_i' \oplus h(N_i \| SID_j)$
$f_i' = h(PWD_i' \oplus Z_i')$		$G_i = h(SID_j \| X_s \| ID_i)$
Checks $f_i' ? = f_i$		$V_2 = GID_i \oplus h(ID_i \| T_1 \| X_s)$
Chooses N_u, timestamp T_1	$\{T_1, GID_i, M_1, B_3, B_4\}$ ----->	$Au_i' = B_4 \oplus h(V_2 \| ID_i)$
Computes $X_s = B_2 \oplus h(PWD_i \| ID_i)$		$token' = Au' \oplus h(SID_j \| ID_i)$
$B_3 = ERD_i \oplus h(T_1 \| X_s)$		$M_1' = h(G_i \| T_1 \| V_2)$
$G_i = Z_i \oplus PWD_i$		Checks $M_1' ? = M_1$
$V_1 = N_u \cdot B_1$, $B_4 = Au_i \oplus h(V_1 \| ID_i)$		Chooses N_s, timestamp T_3
$GID_i = V_1 \oplus h(ID_i \| T_1 \| X_s)$		Computes $SK_j = h(V_2 \| N_s \| ID_i \| SID_j)$
$M_1 = h(G_i \| T_1 \| V_1)$		$M_2 = SK_j \oplus h(X_s \| V_2 \| ID_i)$
		$M_3 = h(G_i \| T_3 \| SK_j)$
Checks $(T_4 - T_3) \le \triangle T$		
Computes $SK_i = M_2 \oplus h(X_s \| V_1 \| ID_i)$		
$M_3' = h(G_i \| T_3 \| SK_i)$	$\{T_3, M_2, M_3\}$ <-----	
Checks $M_3' ? = M_3$		

Fig. 1. Li's Login and Authentication Phase

$V_2' = GID_i \oplus h(ID_i' \| T_1 \| X_s)$
$SK_i' = M_2 \oplus h(X_s \| V_2' \| ID_i') = SK_j'$

3.1.3 User anonymity

Li의 프로토콜은 앞의 서버 키 노출 공격 절에서 분석한 바와 같이 ID_i' 추측과 그 값의 정확성을 식 $M_1' = h(G_i \| T_1 \| V_2)$을 이용하여 검증 가능하다. 재구성한 M_1'은 $h(h(SID_j \| X_s \| ID_i') \| T_1 \| GID_i \oplus h(ID_i' \| T_1 \| X_s))$와 동일하므로 이 식을 사용한다. 결국, 공격자와 악의의 사용자들은 다른 모든 사용자들의 식별자를 계산해낼 수 있고, 이로 인하여 안전한 추적 불가능성을 보장하지 못한다.

3.1.4 Smart-card lost attack

스마트카드 분실 시, 공격자는 카드에 저장된 Au_i와 서버의 SID_j, 추측한 사용자 ID_i'를 이용하여 $token' = Au_i \oplus h(SID_j \| ID_i')$을 계산할 수 있다. 서버의 SID_j는 공격자가 정당한 사용자를 가장하여 쉽게 획득가능하다. 공격자는 사용자를 가장하기 위하여 $T_1, GID_i, M_1, B_3, B_4$를 작성해야 하고, 정당한 사용자로 등록한 공격자나 악의를 가진 사용자는 서버의 비밀키를 알고 있으므로 다른 사용자를 가장하기 위한 모든 정보를 다음 과정을 통해 계산할 수 있다.

1. $h(V_1 \| ID_i)' = Au_i \oplus B_4$ 계산
2. V_1은 V_2와 동일하므로 V_2와 전송 메시지들을 사용하여 $V_2' \ ?= GID_i \oplus h(ID_i' \| T_1 \| X_s)$를 계산
3. 전송 메시지 M_1의 구성 요소인 G_i는 서버의 식 $G_i' = h(SID_j \| X_s \| ID_i)$을 이용하여 계산. 이 식을 이용하면 추측 공격한 ID_i의 정확성 확인가능

공격자는 앞의 1~3번에 의하여 M_1을 계산할 수 있고, 세션 키 $SK_i = M_2 \oplus h(X_s \| V_1 \| ID_i)$를 위한 계산 항목들을 공격자가 위의 과정을 통해 모두 알고 있다.

3.1.5 Ephemeral secret leakage attack

Li 프로토콜의 세션 키 계산은 $h(V_2 \| N_s \| ID_i \| SID_j)$나 $SK_i = M_2 \oplus h(X_s \| V_1 \| ID_i)$로 계산한다. 정당한 사용자를 가장하여 서버의 비밀키 X_s를 획득한 공격자가 서버의 생성 난수 N_s를 알고 있다고 가정할 경우, V_2는 $V_2 = GID_i \oplus h(ID_i \| T_1 \| X_s)$나 $V_1 = GID_i \oplus h(ID_i \| T_1 \| X_s)$로 계산할 수 있다. 서버의 비밀키 X_s를 손쉽게 획득한 공격자는 SID_j가 높은 엔트로피가 아닐 경우, 자신의 ID_i와 G_i를 알고 있으므로 식 $G_i = h(SID_j \| X_s \| ID_i)$을 이용하여 서버의 식별자 SID_j를 획득할 수 있다. 그러므로 Li 프로토콜은 난수와 같은 일시적인 정보들이 노출된 공격에 안전하지 않다.

2. Design analysis

Li의 프로토콜은 등록을 원하는 새로운 사용자의 ID_i에 대한 타당성을 검증하지 않은 체, 바로 다음 단계를 진행한다. 그러므로 Li의 프로토콜은 동일한 ID_i를 소유한 사용자의 중복성 문제가 잠재한다.

IV. The Proposed Protocol

제안 프로토콜은 서버의 비밀키가 노출되는 Li의 프로토콜의 심각한 문제로 인해 발생하는 여러 문제를 개선하기 위한 새로운 프로토콜을 제안한다. 제안 프로토콜은 사용자 등록 단계, 로그인과 인증 단계, 그리고 패스워드 변경 단계로 구성된다.

4.1 User registration phase

1. 사용자 U_i는 자신의 ID_i와 패스워드 PW_i, 그리고 자신의 생체정보 b_i를 입력한다. 스마트카드는 $Gen(b_i) = (\sigma_i, \tau_i)$과 $PWD_i = h(ID_i \| PW_i \| \sigma_i)$을 계산하여 서버 S_j에 $\{ID_i, PWD_i\}$을 안전하게 전송한다.
2. 서버 S_j는 사용자의 ID_i 타당성을 확인하여 타당할 경우, 난수 N_i, ERD_i를 생성하고 $Z_i = h(SID_j \| X_s \| N_i \| ID_i) \oplus PWD_i$, $B_1 = N_i \cdot P$, $Au_i = h(SID_j \| ID_i \| N_i) \oplus token$을 계산하여, ERD_i, N_i, ID_i는 안전하게 저장하고, $\{SID_j, Z_i, ERD_i, B_1, Au_i\}$는 스마트카드에 저장하여 사용자 U_i에게 안전하게 전달한다.
3. 사용자 U_i는 $f_i = h(PWD_i \oplus Z_i)$와 $E_i = Z_i \oplus h(PWD_i \| \sigma_i)$, $RD_i = ERD_i \oplus h(PW_i \| \sigma_i)$, $TD_i = RD_i \oplus h(PWD_i)$를 계산한 후, E_i, TD_i, f_i를 스마트카드에 추가로 저장하고, Z_i와 ERD_i는 삭제한다. 스마트카드에는 최종적으로 $\{SID_j, E_i, f_i, TD_i, B_1, Au_i\}$가 저장된다.

4.2 Login and authentication phase

1.사용자 U_i가 자신의 ID_i와 패스워드 PW_i, 그리고 자신의 생체정보 b_i를 입력하면, 스마트카드는 $\sigma_i' = Rep(b_i, \tau_i)$, $PWD_i' = h(ID_i \| PW_i \| \sigma_i')$, $Z_i' = E_i \oplus h(PWD_i' \| \sigma_i')$, $f_i' = h(PWD_i' \oplus Z_i')$을 계산하여 카드에 저장된 f_i와 f_i'이 같은 값인지 비교한다. 만약 값이 다를 경우 세션을 종료하고, 동일하면 난수 N_u와 타임스탬프 T_1을 생성하여 $\{T_1, ERD_i, GID_i, B_4, M_1\}$을 서버에 전송한다.

$$RD_i' = TD_i \oplus h(PWD_i)$$
$$ERD_i' = RD_i \oplus h(PWD_i \| PW_i)$$
$$G_i = Z_i \oplus PWD_i,\ V_1 = N_u \cdot B_1$$
$$B_2 = Au_i \oplus h(V_1 \| ID_i \| T_1)$$
$$GID_i = V_1 \oplus h(ID_i \| T_1 \| G_i)$$
$$M_1 = h(T_1 \| ERD_i \| G_i \| V_1)$$

2.메시지를 전송받은 서버 S_j는 타임스탬프 T_1의 타당성 검증을 위하여 $(T_2 - T_1) \leq \triangle T$을 계산한다. 만약 타당한 값이 아니면 세션을 종료하고, 타당한 값이면 ERD_i에 매핑되는 ID_i와 난수 N_i를 검색하여 다음 값들을 계산한다.

$$G_i' = h(SID_j \| X_s \| N_i \| ID_i)$$
$$V_2 = GID_i \oplus h(ID_i \| T_1 \| G_i')$$
$$Au_i' = B_2 \oplus h(V_2 \| ID_i \| T_1)$$
$$token = Au_i' \oplus h(SID_j \| ID_i \| N_i)$$
$$M_1' = h(T_1 \| ERD_i \| G_i' \| V_2)$$

계산한 M_1'과 전송 메시지 M_1이 동일하면 난수 N_s, ERD_i^n, 타임스탬프 T_3을 생성하여 세션 키 SK_j를 계산한 후, $\{T_3, M_2, M_3, Q_i\}$를 사용자에게 전송한다.

$$Q_i = ERD_i^n \oplus h(G_i' \| T_3) \oplus h(ID_i \| V_2)$$
$$SK_j = h(T_1 \| T_3 \| V_2 \| X_s \| N_s \| ID_i \| SID_j)$$
$$M_2 = SK_j \oplus h(G_i' \| V_2 \| ID_i \| T_3)$$
$$M_3 = h(G_i' \| T_3 \| SK_j \| Q_i \| ERD_i^n)$$

3.메시지를 전송받은 사용자는 타임스탬프 T_3의 타당성($(T_4 - T_3) \leq \triangle T$)을 검증하여, 타당한 값이 아니면 세션을 종료하고, 타당한 값이면 세션 키 SK_i, ERD_i^n, M_3'을 계산한다. 계산한 M_3'과 전송 메시지 M_3이 동일하면 사용자는 서버를 인증하고 $RD_i^n = ERD_i^n \oplus h(PW_i \| \sigma_i)$를 계산하여 스마트카드에 저장한다. 만약 두 값이 동일하지 않으면 세션을 종료한다.

$$SK_i = M_2 \oplus h(G_i \| V_1 \| ID_i \| T_3)$$
$$ERD_i^n = Q_i \oplus h(G_i' \| T_3) \oplus h(ID_i \| V_1)$$
$$M_3' = h(G_i \| T_3 \| SK_i \| Q_i \| ERD_i^n)$$

User U_i		*Server* S_j
Inputs ID_i, PW_i, b_i		Checks $(T_2 - T_1) \leq \triangle T$
Computes $\sigma_i' = Rep(b_i, \tau_i)$		Retrieves the ID_i, N_i mapped to ERD_i
$PWD_i' = h(ID_i \| PW_i \| \sigma_i')$		$G_i' = h(SID_j \| X_s \| N_i \| ID_i)$
$Z_i' = E_i \oplus h(PWD_i' \| \sigma_i')$		$V_2 = GID_i \oplus h(ID_i \| T_1 \| G_i')$
$f_i' = h(PWD_i' \oplus Z_i')$		$Au_i' = B_2 \oplus h(V_2 \| ID_i \| T_1)$
Checks $f_i' ? = f_i$	$\{T_1, ERD_i, GID_i, B_2, M_1\}$ -->	$token' = Au_i' \oplus h(SID_j \| ID_i \| N_i)$
Chooses N_u and a timestamp T_1		$M_1' = h(T_1 \| ERD_i \| G_i' \| V_2)$
Computes $RD_i' = TD_i \oplus h(PWD_i)$		Checks $M_1' ? = M_1$
$ERD_i' = RD_i' \oplus h(PWD_i \| PW_i)$		Chooses N_s, ERD_i^n and a timestamp T_3
$G_i = Z_i \oplus PWD_i$, $V_1 = N_u \cdot B_1$,		Computes $Q_i = ERD_i^n \oplus h(G_i' \| T_3) \oplus h(ID_i \| V_2)$
$B_2 = Au_i \oplus h(V_1 \| ID_i \| T_1)$		$SK_j = h(T_1 \| T_3 \| V_2 \| X_s \| N_s \| ID_i \| SID_j)$
$GID_i = V_1 \oplus h(ID_i \| T_1 \| G_i)$		$M_2 = SK_j \oplus h(G_i' \| V_2 \| ID_i \| T_3)$
$M_1 = h(T_1 \| ERD_i \| G_i \| V_1)$		$M_3 = h(G_i' \| T_3 \| SK_s \| Q_i \| ERD_i^n)$
Checks $(T_4 - T_3) \leq \triangle T$		
Computes $SK_i = M_2 \oplus h(G_i \| V_1 \| ID_i \| T_3)$		
$ERD_i^n = Q_i \oplus h(G_i' \| T_3) \oplus h(ID_i \| V_1)$	<-- $\{T_3, M_2, M_3, Q_i\}$	
$M_3' = h(G_i \| T_3 \| SK_i \| Q_i \| ERD_i^n) ? = M_3$		
Saves $RD_i^n = ERD_i^n \oplus h(PW_i \| \sigma_i)$		

Fig. 2. Proposed Login and Authentication Phase

4.3 Password change phase

패스워드 변경을 원하는 사용자는 다음과 같은 패스워드 변경단계를 진행한다.

1.스마트카드는 사용자 U_i가 자신의 ID_i와 PW_i, 생체정보 b_i를 입력하면, $\sigma_i' = Rep(b_i, \tau_i)$, PWD_i, Z_i, f_i'을 계산하여 카드에 저장된 f_i와 계산한 f_i'이 같은 값인지 비교한다. 만약 다른 값일 경우, 세션을 종료하고, 그렇지 않으면 다음 과정을 진행한다.

$$PWD_i' = h(ID_i \| PW_i \| \sigma_i')$$
$$Z_i = E_i \oplus h(PWD_i' \| \sigma_i')$$
$$f_i' = h(PWD_i' \oplus Z_i')$$

2. 사용자는 새로운 패스워드 PW_i^n을 입력한다.

3.스마트카드는 다음 값들을 계산한 후, 새로운 E_i^n, f_i^n, TD_i^n을 각각 저장한다.

$$PWD_i^n = h(ID_i \| PW_i^n \| \sigma_i^n)$$
$$Z_i^n = Z_i \oplus PWD_i' \oplus PWD_i^n$$
$$f_i^n = h(PWD_i^n \oplus Z_i^n)$$
$$E_i^n = Z_i^n \oplus h(PWD_i^n \| \sigma_i)$$
$$RD_i = TD_i \oplus h(PWD_i)$$
$$ERD_i = RD_i \oplus h(PW_i \| \sigma_i)$$
$$RD_i^n = ERD_i \oplus h(PWD_i^n \| \sigma_i)$$
$$TD_i^n = RD_i^n \oplus h(PWD_i^n)$$

V. Analysis of the Proposed Protocol

본 장에서는 제안 프로토콜과 관련 프로토콜들을 비교·분석하여, 제안 프로토콜의 안전성, 그리고 계산 비용과 통신 비용에 대한 효율성을 제시한다.

1. Informal security analysis

4.1.1 Perfect forward secrecy

제안 프로토콜의 세션 키 SK_i와 SK_j의 계산은 각각 $M_2 \oplus h(G_i \| V_1 \| ID_i \| T_3)$과 $h(T_1 \| T_3 \| V_2 \| X_s \| N_s \| ID_i \| SID_j)$로 계산하므로 공격자가 서버의 비밀키를 안다고 할지라도 N_s, G_i V_2를 알아야 한다. V_2를 알기 위해서는 G_i를 알아야 하므로 결국, 서버의 비밀키 외에 난수 N_i를 알아야 한다. 또는 사용자측의 Z_i와 PWD_i를 알면 G_i를 계산할 수 있으나, PWD_i는 저장하지 않는 값이고, Z_i를 알려면 스마트카드의 저장정보 E_i와 저장하지 않는 값 $h(PWD_i \| \sigma_i)$ 를 알아야 한다. 그러므로 공격자가 서버의 비밀키를 획득하였다 할지라도 다른 값들이 노출되지 않는 한 제안 프로토콜은 안전하다.

4.1.2 Smart-card lost attack

공격자가 스마트카드에 저장된 E_i와 f_i를 이용하여 사용자의 PW_i를 획득하려면 Z_i와 PWD_i, 그리고 사용자의 생체정보 b_i를 알아야 한다. 그러나 이 정보들은 스마트카드에 저장하지 않는 값이므로 스마트카드로부터 PW_i를 계산해내기 어렵다. 스마트카드를 획득한 공격자는 카드의 Au_i와 전송 메시지 B_2를 계산하여 $h(V_1 \| ID_i \| T_1)$ 값을 얻을 수 있다. 그러나 해시연산한 값으로부터 높은 엔트로피의 V_1을 계산해내기 어렵다. 또한 V_1은 매번 새로운 난수 N_u를 생성하여 $V_1 = N_u \cdot B_1$을 계산하므로 임의 세션들간의 연관성을 찾기 어렵다. 또한, 공격자가 전송 메시지 ERD_i와 스마트카드의 TD_i를 이용하여 PW_i를 획득하려면 $h(PWD_i \| PW_i)$ 값을 알아야 한다. 그러나 PW_i와 PWD_i는 저장하지 않는 값이며, 다른 계산과정을 통해 $h(PWD_i \| PW_i)$ 값 자체를 획득할 수 있는 방법도 전무하다. 또한, ERD_i는 매 세션마다 다른

Table 2. Comparison Security and Design Defect

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
Lee et al.[2]	X	X	X	O	X	X	O	O	O	X	X
Chang et al.[4]	O	X	X	O	O	O	O	X	X	X	X
Sahoo et al.[7]	X	X	O	O	X	O	X	X	O	O	X
Mirsaraei et al.[8]	X	X	X	X	O	O	O	O	X	O	X
Li[9]	X	X	X	X	X	O	X	O	X	O	X
Arpitha et al.[6]	X	O	O	O	O	O	O	O	X	O	X
Zargar et al.[11]	X	X	O	X	X	X	O	O	X	O	X
Proposed	O	O	O	O	O	O	O	O	O	O	O

F1:User anonymity, F2:User un-traceability, F3:Forward secrecy, F4:Smart-card lost attack, F5:User impersonation attack, F6:Offline password guessing attack, F7:Server impersonation attack, F8:Privileged insider attack, F9:Ephemeral secret leakage attack, F10:Replay attack, F11:Design defects

난수를 사용하기 때문에 이와 관련된 RD_i와 TD_i도 매번 변경되어 세션 간의 연관성을 찾기 어렵다. 스마트카드의 저장정보 B_1은 $N_i \cdot P$와 같이 계산하기 때문에 ECDHP의 안전도에 의하여 공격자는 난수 N_i을 계산해내기 어렵다.

4.1.3 Privileged insider attack

사용자가 서버에 제출한 ID_i와 PWD_i를 내부 공격자가 획득하였다고 가정할 경우, PWD_i는 $h(ID_i||PW_i||\sigma_i)$와 같이 계산하므로 높은 엔트로피의 생체정보 σ_i때문에, 해시 처리된 PWD_i로부터 사용자의 패스워드 PW_i를 추측 공격하기 어렵다.

4.1.4 Ephemeral secret leakage attack

제안 프로토콜의 세션 키는 $SK_j = h(T_1||T_3||V_2||N_s||ID_i||SID_j)$와 같이 계산한다. 그러므로 난수 이외에 각 객체의 식별자를 공격자가 모두 알고 있다고 하더라도 서버의 비밀키 X_s와 V_2를 알아야 만이 세션 키를 계산할 수 있고, N_s와 V_2는 매 세션마다 다른 값들이기 때문에 세션 간의 연관성을 추적하는 것이 어렵다.

4.1.5 User anonymity

제안 프로토콜은 사용자 ID_i를 전송 메시지로 사용하지 않으며, ERD_i는 매 세션마다 다른 난수를 사용하여 세션 간의 연관성을 찾기 어렵다. 그러므로 ERD_i 값이 노출된다고 하더라도 매번 다른 난수 ERD_i로 변경되어 익명성뿐만 아니라 추적 불가능성도 함께 제공한다.

2. Computational Evaluation

본 절의 계산 비용에서 사용한 해시함수 T_h의 실행시간은 0.002486ms, 타원곡선 상의 곱셈 연산 T_m은 1.031757ms, 비밀키 암호방식의 암·복호화 T_s는 0.0046ms, 퍼지 추출함수 T_f는 T_m 연산의 실행시간과 동일한 것으로 한다. Table 3은 제안 프로토콜과 관련 프로토콜들과의 계산 비용을 분석한 것으로, 로그인과 인증단계의 총 실행시간은 Fig. 3과 같다.

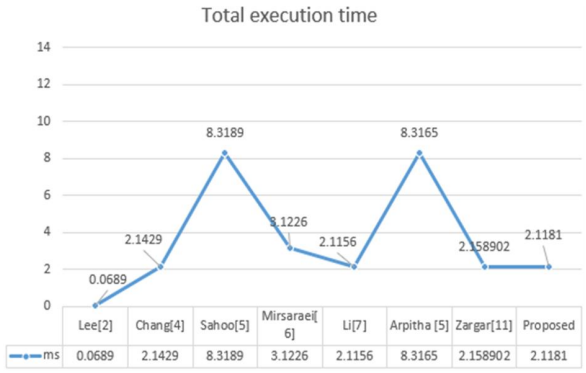

Fig. 3. Comparison of Execution Time

Arpitha 등의 프로토콜은 T_m의 연산이 누락된 잘못된 계산 결과를 나타내었고, 본 논문에서는 이러한 오류를 바로잡아 계산 비용을 분석하였다. Li의 프로토콜은 2.1156ms, 다음으로 빠른 프로토콜은 2.1181ms의 제안 프로토콜이다. 제안 프로토콜의 실행시간은 Li의 프로토콜에 비하여 0.0025ms만 증가하여 실행시간의 증가가 매우 작은 것을 알 수 있다. 또한, 제안 프로토콜은 최근에 제안된 경량의 Zargar et al.[10] 프로토콜보다 더 빠른 것을 알 수 있다.

Table 4는 전송 비용을 분석한 결과로, 타원곡선 포인트의 길이는 512비트, 타임스탬프의 길이는 64비트, 나머지 항목들은 128비트로 가정하였고, 관련 프로토콜 중 가장 적은 전송 비용은 1,280비트로, Li의 프로토콜, Sahoo, Mirsaraei, Arpitha 등의 프로토콜이고, 그다음으로 적은 전송 비용은 1,408비트의 제안 프로토콜이다.

Table 3. Comparison of the Computational Cost

	User	Server	RC	Total Computational Cost
Lee et al.[2]	$9T_h$	$7T_h$	$8T_h+2T_s$	$24T_h+2T_s$
Chang et al.[4]	$9T_h+1T_m+1T_{fc}$	$7T_h$	$15T_h+1T_m$	$31T_h+2T_m+1T_{fc}$
Sahoo et al.[7]	$8T_h+2T_s+3T_m+1T_f$	$4T_h+2T_s+2T_m$	$3T_h+2T_s+2T_m$	$15T_h+6T_s+7T_m+1T_f$
Mirsaraei et al.[8]	$5T_h+1T_m+1T_f$	$6T_h+1T_m$	-	$11T_h+2T_m+1T_f$
Li[9]	$10T_h+1T_m+1T_f$	$11T_h$	-	$21T_h+1T_m+1T_f$
Arpitha et al.[6]	$5T_h+3T_m+2T_s+1T_f$	$4T_h+2T_m+2T_s$	$5T_h+2T_m+2T_s$	$14T_h+7T_m+6T_s+1T_f$
Zargar et al.[10]	$5T_h+1T_m+1T_H$	$7T_h+1T_m$	-	$13T_h+2T_m1T_H$
Proposed protocol	$12T_h+1T_m+1T_f$	$10T_h$	-	$22T_h+1T_m+1T_f$

Table 4. Comparison of the Communication Costs

	Communication Cost	Number of messages
Lee et al.[2]	1,920 bits	4
Chang et al.[4]	2,176 bits	4
Sahoo et al.[7]	1,280 bits	4
Mirsaraei et al.[8]	1,280 bits	2
Li[9]	1,280 bits	2
Arpitha et al.[6]	1,280 bits	3
Zargar et al.[10]	1,664 bits	2
Proposed protocol	1,408 bits	2

VI. Conclusions

본 논문에서는 Li의 프로토콜에 대한 보안 분석과 설계상의 문제를 분석하였고, 분석 결과, 해당 프로토콜은 모든 사용자들에게 서버의 비밀키가 그대로 노출되는 심각한 취약점을 가지고 있었다. 이로 인해 스마트카드 분실 공격, 전방향 안전성, 세션 키 노출 문제, 임시 정보 노출 공격, 사용자 익명성 침해 등의 문제가 발생할 수 있음을 확인하였다. 본 논문에서는 익명성을 보장하면서 여러 공격에 대한 저항성을 갖는 사용자 인증 프로토콜을 제안하였다. 안전성 분석 결과, 제안 프로토콜은 스마트카드 분실 공격, 서버 사칭 공격, 세션 키 유출 문제, 오프라인 패스워드 추측 공격 등에 대해 안전하며 전방향 안전성과 추적 불가능성 역시 만족함을 보였다. 아울러 성능 평가에서는 향상된 보안성에도 불구하고 짧은 실행시간과 전송 비용 증가가 크지 않음을 확인하였다. 따라서 제안 프로토콜은 안전성과 효율성을 동시에 만족하는 실용적인 사용자 인증 프로토콜이라 할 수 있다.

REFERENCES

[1] P. K. Dhillon and S. Kalra, "Secure multi-factor remote user authentication scheme for Internet of Things environments," International Journal of Communication Systems, Vol. 30, No. 16, pp. e3323, Apr. 2017. DOI: 10.1002/dac.3323

[2] H. J. Lee, D. W. Kang, J. Y. Ryu, Y. S. Lee, "A three-factor anonymous user authentication scheme for Internet of Things environments," Journal of Information Security and Applications 52, Vol. 52, No. 6, pp. 1-14, 102494, 2020.

[3] A. Li, B. Kang, Y. Huo, X. Zuo, and S. Niu, "Analysis and Improvement on a Three-Factor Authentication Scheme in IoT Environment," Vol. 4, No. 2, pp. 81-89, June 2023. DOI: 10.54097/fcis.v4i2.10301

[4] Y. F. Chang, W. L. Tai, P. L. Hou and K. Y. Lai, "A Secure Three-Factor Anonymous User Authentication Scheme for Internet of Things Environments," Symmetry 2021, 1121. Vol. 13, No. 7, pp. 1-17, 2021 DOI:10.3390/sym13071121

[5] N. W. Lo, C. Y. Chuang, J. J. Huang, Y. X. Luo, "Authentication protocol for vehicular networks using Zero-Knowledge Proofs and Elliptic Curve Cryptography," ICT Express 2025. pp. 1-7, June 2025, DOI: 10.1016/j.icte.2025.04.014

[6] T. Arpitha, D. Chouhan, and J. Shreyas, "Anonymous and robust biometric authentication scheme for secure social IoT healthcare applications," Journal of Engineering and Applied Science, Vol. 46, No. 2, pp. 1-13, Jan. 2024. DOI: 10.1186/s44147-023-00342-1

[7] S. S. Sahoo, S. Mohanty, and B. Majhi, "A secure three factor based authentication scheme for health care systems using IoT enabled devices," Journal of Ambient Intelligence and Humanized Computing, 2021, Vol. 12, No. 1, pp. 1419-1434, 2021. DOI: 10.1007/s12652-020-02213-6

[8] A. H. G. Mirsaraei, A. Barati, and H. Barati, "A secure three-factor authentication scheme for IoT environments," Journal of Parallel and Distributed Computing, Vol. 169, No. 1, pp. 87-105, June 2022. DOI: 10.1016/j.jpdc.2022.06.011

[9] Y. Li, "A secure and efficient three-factor authentication protocol for IoT environments," Journal of Parallel and Distributed Computing, Vol. 179, Article 104714, pp. 1-16, Sep. 2023. DOI: 10.1016/j.jpdc.2023.104714

[10] G. R. Zargar, H. Barati, A. Barati, "An authentication mechanism based on blockchain for IoT environment," Cluster Computing (2024), Vol. 27, No. 16, pp. 13239-13255, 2024, DOI: 10.1007/s10586-024-04565-6

Authors

Mi-Og Park received the M.S. and Ph.D. degrees in Computer Science and Engineering from Soongsil University, Korea, in 1993 and 2004, respectively. Dr. Park joined the faculty of the Department of Computer Engineering at Sungkyul University, Korea, in 2005. She is interested in mobile security, security protocol and IoT security.