SPAS: An Authentication Scheme to Prevent Unauthorized Access of Information from Smart Card

Ajay Kumar Sahu¹* and Ashish Kumar²

¹Department of Computer Science and Engineering, Raj Kumar Goel Institute of Technology and Management, Ghaziabad 201004, Uttar Pradesh, India
²Department of Computer Science and Engineering, ITS, Greater Noida 201306, Uttar Pradesh, India

ABSTRACT

Nowadays internet has become indispensable part of one’s life. Therefore, security and privacy are of critical concern to retain user’s confidence in network services and applications. Several password verification based schemes/protocols have been used for authentication over insecure channel to protect resources from unauthorized access in networked environment. However, the schemes were not fault tolerant. Also, the feasibility for implementation in some of the applications was questionable. Therefore, we have devised a scheme SPAS (Secure and Provable Authentication Scheme) to overcome the issues prevalent in existing schemes. The objective was to keep the computational and communication cost low. The analysis of the presented scheme SPAS over existing schemes corroborates its effectiveness in tackling various attacks and uniqueness. Further, the performance analysis of the presented scheme is also given to strengthen the proposal.

Keywords: Hash function, identity; security, information retrieval, key agreement, mutual authentication, password, smart card

INTRODUCTION

Growing usage and applications of technology in all possible domains of human lives is posing a challenge. It has become hard to authenticate the user. Many online frauds are reported in the literature accounted for significant financial loss.
Consequently, the need arises to authenticate the genuine users over an insecure channel like Internet. The most commonly used technique is password based authentication protocol (ElGamal, 1985; Kocher et al., 1999; Lamport, 1981; Tang et al., 2002). The password based techniques are prone to various attacks due to human cognitive capability of designing and remembering complex passwords. Using a smart card, having various advantages like portability, low cost and availability. By using the smart card we can practically and efficiently implement these password based authentication protocols easily (Chang & Wu, 1991; Chien et al., 2002; Hwang & Li, 2000; Ku & Chen, 2004; Messerges et al., 2002; Yang & Shieh, 1999) easily. Several protocols considering static identity have been proposed in the literature (Karuppiah & Saravanan, 2014; Karuppiah & Saravanan, 2015; Kumar et al., 2011; Song, 2010; Wang et al., 2014). Static identity based protocols are easier targets for the attackers. Recently, researchers have designed schemes (Chang & Chang, 2009; Chang et al., 2013; Das et al., 2004; Khan et al., 2014; Li et al., 2014; Wen & Li, 2011; Wang et al., 2009) based on dynamic identity to manage the static identity problem. However, the schemes suffers with high storage and computation cost. In some cases they were shown to be unsafe. Kumari et al. (2014), stated that Chang et al. (2013) protocol was susceptible to various attacks like impersonation, password guessing and anonymity violation. Later suggested an enhanced protocol (“An improved remote user authentication scheme with key agreement”) to overcome existing security and identity concerns. However, analysis of the protocol proposed by Kumari et al. (2014) revealed that it violated on certain aspects i.e. chip/smart card loss and user anonymity and compromised the offered safety against various attacks. In this protocol attacker may leak the privacy and can steal server’s secret key as well as password of the authenticate user, which may impact the entire system. Therefore, we have designed and proposed an efficient, Secure and Provable Authentication Scheme (SPAS), to overcome the observed security weaknesses.

The remaining section in this paper is structured in a systematic way as follows: Notations, System design and Capabilities of adversary are shown in as part 2; we proposed SPAS (Secure and Provable Authentication Scheme) in part 3; Security analysis of suggested scheme is illustrated in part 4; Performance evaluation is shown in part 5 and lastly, conclusion of our scheme is discussed.

NOTATIONS, SYSTEM DESIGN AND CAPABILITIES OF ADVERSARY
In this section, initially basic notations are described then demonstrate system design and capabilities of adversary being used in the scheme.

Notations and Description
The following symbols/notations are preferred in this paper as described in Table 1.
Table 1
Symbols/Notations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_i$</td>
<td>Remote user</td>
</tr>
<tr>
<td>$s_i$</td>
<td>System server</td>
</tr>
<tr>
<td>$sc_i$</td>
<td>Smart Card</td>
</tr>
<tr>
<td>$id_i$</td>
<td>User’s identity</td>
</tr>
<tr>
<td>$c_{id_i}$</td>
<td>User’s variable identity</td>
</tr>
<tr>
<td>$pw_{id_i}$</td>
<td>User’s password</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Random number chosen from user</td>
</tr>
<tr>
<td>$mp_{\alpha}$</td>
<td>Encrypted/modified password</td>
</tr>
<tr>
<td>$\beta_i$</td>
<td>Random number chosen by Server for user</td>
</tr>
<tr>
<td>$x_{1, x_2}$</td>
<td>Server’s private key and Server’s secret number</td>
</tr>
<tr>
<td>$h(.)$</td>
<td>Hash operation</td>
</tr>
<tr>
<td>$| |$</td>
<td>Concatenation process</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>XOR operation</td>
</tr>
<tr>
<td>$t_1$</td>
<td>User’s current timestamp</td>
</tr>
<tr>
<td>$t_2$</td>
<td>Server’s current timestamp</td>
</tr>
<tr>
<td>$\Delta t$</td>
<td>maximum transmission delay time</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Random Number Chosen by smart card</td>
</tr>
<tr>
<td>$n$</td>
<td>Number of times user registers in case of smart card lost</td>
</tr>
<tr>
<td>$Z$</td>
<td>Adversary/Attacker</td>
</tr>
</tbody>
</table>

System Design
Our scheme includes five fundamental stages as described in section 3. Initially, user’s information is fed to the terminal and forwarded to the server for registration. Thereafter, server delivers the chip card to remote user along-with security parameters. The registration phase is required only once in our scheme unless and otherwise required re-registration in unavoidable conditions. For accessing resources, permission is given by the server once the credentials are verified in Login stage. The communication takes place only after both, the server and user validates each other. The login and authentication process is usually done several times. In the scheme, a two factor authentication process is used to ensure that chip card owner can be allowed the accessing of the server. Password update and chip card loss re-registration stages are the very useful approaches to overcome the security issues in our proposed scheme.
Capabilities of the Adversary

According to various researches (Chang & Chang, 2005; Wang et al., 2014) of two-factor authentication, the adversary Z has the following capabilities:

In the open communication channel, the adversary Z having full control as it can intercept, remove, update or again send the eavesdropped messages over open communication media.

Adversary Z can get the password of an authenticate user by an infected card reader or obtain the smart card parameters but both cannot be achieved simultaneously.

Adversary Z can also obtain the servers secret key in the case of forwarding secrecy.

THE PROPOSED AUTHENTICATION SCHEME: SPAS

The working of proposed SPAS (Secure and Provable Authentication Scheme) is shown in Figure 1. It consists of five stages namely: 1. Registration stage, 2. Login stage, 3. Verification stage, 4. Password update stage and 5. Re - registration stage. The SPAS is a simple, adequate and most secure based on variable identity authentication scheme to prevent those attacks that exist in the scheme (Kumari et al., 2014). To reduce storage cost, computing complexity as well as to maintain its high performance along with certified security, we need OR, Ex-OR, and elementary hash operations in our scheme.

Within registration stage, the smart card contains data which is dependant on the information produces through user and several credentials inserted through server. In the login/verification phase, both the user and server authenticate each other and after successful mutual authentication, a session key is established before communication between them as various schemes (Kumari & Khan, 2013; Tu et al., 2015; Lu et al., 2015). The initial validity is checked by the smart card then sends user’s information towards server for more verification. If the authentication of the server or user fails, this is recommended that the login stage is denied.

User’s Registration Stage

In the beginning of this phase u, registers/re-registers with s, whenever want some services from it. Assume \( x_1 \) and \( x_2 \) are private keys and a secret number of the \( s_i \). Suppose \( n \) specifies how many times a user registers by authentication server \( s_i \). In mishap-penning cases like chip card loss, theft or snatched, the chip card can revoke through applying the value of \( n \) and value of \( n \) is saved in a database of user’s history on \( s_i \). The authentication server stores these secret key \( x_1 \) and number \( x_2 \) securely. The entire registration phase having a number of computation steps as follows:
SPAS: An Authentication Scheme to Prevent Unauthorized Access of Information from Smart Card

Stage 1: Registration – User’s

Stage 2: Login – User’s login for system allocation

Stage 3: Mutual Verification - Server and User validate to one another

Stage 4: Password update – Give services for password modification of smart card at any given time

Stage 5: Revocation - Provide the facility for re-registering if smart card misplaced or stolen

Figure 1. Methodology Used

- User \( u_i \) select \( id_i, pwd_i \) and an arbitrary number \( \alpha \).
- Compute \( mpw_id_i = h(\alpha||pwd_i) \) then transfers \( \{id_i, mpw_id_i\} \) towards \( s_i \).
- The registration credentials of user are verified by the server. If selected identity is matched with another in the database, \( s_i \) warns \( u_i \) to select another \( id_i \).
- Set \( n=0 \) by \( s_i \) for unique \( u_i \), moreover, specify \( n=1 \) by \( s_i \) for re-registering user’s into the server. Increment \( n \) each time of re-registration then \( id_i \) will be stored in the database.
- After getting \( \{id_i, mpw_id_i\} \), server selects a random number \( \beta_i \), which is different for each user.
- Server computes \( A_i = h(id_i||\beta_i||mpw_id_i) \), where \( id_i = (id_i||n) \), \( B_i = h(h(id_i)||x_i) \oplus mpw_id_i \), \( C_i = \beta_i \oplus h(h(id_i)||x_i) \oplus mpw_id_i \) and \( D_i = \beta_i \oplus h(x_i||x_i) \).
- Stores \( \{C_i, A_i, D_i, h(.)\} \) into chip card and deliver \( \{chip card, B_i\} \) to \( u_i \).
- \( u_i \) computes \( E_i = h(id_i||pwd_i) \oplus \alpha, F_i = B_i \oplus \alpha \) and stores \( \{E_i, F_i\} \) into chip card.

User’s Login Stage

For obtaining services from \( s_i \), a user must login into the server. For this, it must insert its personal chip card into device then insert own \( id_i \) as well as \( pwd_i \), further:
• Compute \( \alpha = E \odot h(id)p_d, mp_d = h(a)p_d, h(h(id)lx_i) = F \odot mp_d \odot \alpha, \beta = C \oplus h(id)lx_i \odot mp_d. \)
• Smart card checks \( A, ? = h(id)\beta \odot mp_d \) i.e it is correct or not.
• If not correct, chip card drops this session. If correct, computes \( h(x_i) = \beta \odot D, B_i = F \odot \alpha. \)
• Smart card acquires current time-stamp \( t_i, \) computes \( c, d_i = h(id) \odot h(B_i)\beta \odot t_i, B_i = B_i \odot h(\beta \odot t_i), G_i = B_i \odot mp_d. \)
• Select arbitrary \( \gamma, \) compute \( H_i = h(id)\gamma, I_i = G_i \odot H_i, J_i = h(B_i)\beta \odot G_i \odot H_i, K_i = \beta \odot (h(x_i)lx_i) \odot t_i. \)
• Transmits \( \{ c, d_i, B_i, J_i, K_i, t_i, I_i \} \) to server via public channel.

**Verification Stage**

After getting login request message \( \{ c, d_i, B_i, J_i, K_i, t_i, I_i \} \) from \( u_i, \) the server authenticates the user and after proper mutual authentication, session key will be established as follows:

• Server set current time instant \( t_s, \) verify \( t_s \) is authentic or not, means \( t_s - t \leq \Delta t. \)
• If \( t_s \) is not correct, server denies all the login request and drop this session.
• For obtaining the value of \( n \) in database, determine \( id_n = (id)n. \)
• If timestamp \( t_i \) is valid, server compute \( \beta = K \odot (h(x_i)lx_i) \odot t_i, B_i = B_i \odot h(\beta \odot t_i), h(id_i) = c, d_i \odot h(B_i)\beta \odot t_i, G_i = h(id_i)lx_i), H_i = h(\beta \odot (h(x_i)lx_i) \odot t_i). \)
• Verify equation \( J_i = h(B_i)\beta \odot H_i \odot \beta \odot t_i \) holds or not.
• If true, server acquires current timestamp \( t_j, \) compute \( a = h(G_i)\beta \odot t_j \), transmits \( \{ a, t_j \} \) towards the user.
• After getting \( \{ a, t_j \} \) from the server, smart card verifies validity of \( t_j. \)
• If timestamp \( t_j \) is true, check the equation \( a = h(G_i)\beta \odot t_j \) holds or not. If correct, both \( u_i \) and \( s \) mutual validate to one another otherwise, will be aborted by the server.
• Server and user agreed upon a common session key. Compute session key for the user is \( s = h(G_i)\beta \odot t_j \odot h(x_i)lx_i) \odot H_j. \)

**Password Update Stage**

Considering security parameters, if any user desires to modify its own password \( p_d \), with new password \( p_d' \), in the system, user inserts its own chip card into a device of chip card reader then input its own \( id \) and \( p_d \). The following computation has been performed by the smart card without the involvement of remote server \( S. \)

• Smart card computes \( \alpha = E \odot h(id)p_d, mp_d = h(a)p_d, h(h(id)lx_i) = F \odot mp_d \odot \alpha, \beta = C \odot h(id)lx_i \odot mp_d. \)
• Verify equation \( A, ? = h(id)\beta \odot mp_d \) is correct or not.
• If true, a user is permitted to modify his \( pwd \) otherwise session is aborted.
SPAS: An Authentication Scheme to Prevent Unauthorized Access of Information from Smart Card

- Smart card computes \(mp \oplus d_{new} = h(a \| p \| d_{new})\), \(E_{i, new} = h(id \| p \| d_{new}) \oplus a\), \(F_{i, new} = F_{i} \oplus mp \oplus d_{new}\), \(C_{i, new} = C_{i} \oplus mp \oplus d_{new}\), \(A_{i, new} = h(id \| \beta \| mp \| d_{i, new})\).
- Replaces old \(\{E_{i}, A_{i}, F_{i}, C_{i}\}\) with \(\{E_{i, new}, A_{i, new}, F_{i, new}, C_{i, new}\}\).
- Hence, modified password has changed successfully and session is terminated.

Revocation Stage of Lost Smart Card

When any user misplaces smart card then it forwards an application towards server for its revocation. At that moment server asks some credentials from the user to check the authenticity, like Adhaar number, Mobile OTP, Birth date, a card number of identity proof, Voter identity, Name of mother’s maiden or any other user’s known value. Subsequently after checking the validity of the revocation request generating by user server updates existing value of \(n\) for re-registering the chip card. In each time of misplaced or lost smart card case, \(n\) is incremented by 1. Then after, user re-registers with the server without updating its own identity. For the revocation/re-registration of a card, it is expected from a customer that does not avail any earlier values like earlier password, an arbitrary number otherwise by availing same values which are already stored within misplaced or stolen smart card anybody may masquerade as a server’s legitimate user.

SECURITY ANALYSIS OF PROPOSED SCHEME

Here we have analyzed SPAS scheme and shown that this scheme is robust and secure against various attacks:

Provide User’s Un-traceability and Anonymity

In the scheme, it is hard to track privacy/identity and the claim is corroborated by the analysis of the scheme.

- When attacker \(Z\) receives \(\{E_{z}, F_{z}, C_{z}, A_{z}, D_{z}\}\) parameters from smart card then calculate \(a_{z} = E_{z} \oplus h(id \| p \| d_{z})\), \(mp \oplus d_{z} = h(a_{z} \| p \| d_{z})\), \(h(id_{z}) = F_{z} \oplus mp \oplus d_{z} \oplus a_{z}\), \(\beta = h(id_{z}) \oplus C_{z} \oplus mp \oplus d_{z}\), \(h(x_{z}) = \beta \oplus D_{z}\). Hence, attacker can achieve the parameter value of \(h(x_{z})\).
- Any login message as \(\{c, d_{i}, B_{i}, J_{i}, K_{i}, t_{i}, l_{i}\}\) intercepted by attacker can calculate these parameters like \(\beta = K_{i} \oplus h(x_{i} \| l_{i})\), \(B_{i} = B_{i} \oplus h(\beta \| l_{i})\), \(h(id) = c_{i} \oplus h(B_{i} \| l_{i})\). As a result, an attacker can obtain the \(h(id)\) in place of original \(id\) of any legal end user. Hence, confirm that the scheme maintain user’s un-traceability and anonymity.

Resist Off-line Password Assumption Attack

In such type of violation, attacker may guess password by applying user’s smart card in offline manner. Here, attacker may arrange smart card of any user from either stolen/ misplaced, then compute the following:
An attacker can obtain the value of $\alpha_i$ from above section then compute $E_i \oplus \alpha_i = h(id_i \| p_w d_i)$.

From $h(id_i \| p_w d_i)$, the attacker cannot obtain the original password without knowing $id_i$ of the user.

**Prevent Chip Card Loss/Misplaced Attack**

Suppose adversary get hold of the chip card of legal user and rival succeeds in obtaining entire information, in such case the scheme will show that $Z$ cannot get favourable information. From the above two sections, anyone can recognize values of {$h(id_i $), $B_i$, $\beta_i$, $\alpha_i$} from attacks but not {$id_i$, $p_w d_i$}, so $G_i = h(h(id_i \| x_1))$ cannot be computed and $x_1$ (secret key) cannot be known. Further, the value of parameter of $I_i = G_i \oplus H_i$ cannot be computed without $G_i$. Suppose at any time the server receives the message {$c, d_i$, $B_i'$, $J_i^*$, $K_i$, $t_1$, $I_i^*$} from attacker, but the server will not accept this login request as $J_i^* \neq J_i^*$ for different $G_i$. Hence, our scheme can prevent the chip card loss/misplaced attack.

**Prevent User Mask and Server Pose Violation**

Any attacker $Z$ needs the correct login request {$c, d_i$, $B_i'$, $J_i$, $K_i$, $t_1$, $I_i$} if willing to mask an authenticate customer for the server. As described in previous section, no attacker can satisfy the authentication equation of $J_i^* = J_i^*$ successfully without $G_i^*$. Hence, the suggested scheme avoids user mask violation. Assume that, any adversary computes the valid value of $a = h(G_i^* \| \beta_i \| t_1)$, then masquerading attack is possible in server but attacker cannot obtain correct value of $a$ as confidential key $x_1$ or $G_i$ cannot be achieved. Hence, suggested scheme prevents server posed violations.

**Prevent Forward Secrecy**

This is used as very helpful tool for providing secure information to users. Here, with the help of following parameters {$h(id_i \| x_1)$, $\beta_i$, $h(x_2 \| x_1)$, $h(h(id_i \| x_1))$}, session key is obtained by adversary as: $s_k = h(h(id_i \| x_1) \| \beta_i \| t_1 \| h(x_2 \| x_1) \| h(h(id_i \| x_1)))$ that means it is necessary for attacker to know the parameter value of $h(h(id_i \| x_1))$. In our scheme, if an attacker can obtain {$h(id_i \| x_1)$, $\beta_i$, $h(x_2 \| x_1)$, $h(id_i)$}, but cannot obtain the number $\gamma$ anyhow, since it is computed arbitrary in each session. Hence, early session keys cannot be achieved by adversary accurately.

**Prevent Replay Violation**

To prevent replay attack, we use the concept of current timestamps in scheme. When server receives the request of login message as in the form {$c, d_i$, $B_i'$, $J_i$, $K_i$, $t_1$, $I_i$}, the validity of timestamp $t_1$ is checked by server immediately. Likely, the response message is received by user as {$a$, $t_3$} from server, then verify authenticity of $t_3$ firstly. If time instant was not
appropriate, login request messages will be denied by both user and server. So, the suggested scheme prevents replay violation.

**Resist Conspirator Attack**

In view of the suggested idea, user transfers modified password in place of plain text. A random number $\alpha$ is used by the user for protecting password against conspirator's attack and compute the value of modified password as $mp_wd_i = h(\alpha \| p_wd_i)$. Here, attacker doesn't know the value of $\{\alpha, p_wd_i\}$ both, and cannot guess both of them simultaneously in polynomial time. Hence, in the scheme, there is no opportunity for prediction of possible password and check its prediction is correct. So, the scheme prevents from insider attack.

**Maintain Mutual Verification**

The suggested idea describe that validity of legal user verified through server by checking the validity of the equation $J_i = h(Bi \| Hi \| t_1)$. In the same way, the user check the authenticity of the server by verifying the equation $a_3^* = h(Gi \| Hi \| t_3)$. In this manner to provide proper and secure communication both server and user validate to each other. Hence, suggested scheme must maintain mutual verification.

**Provide Session Key Establishment**

Here, session key $s_k = h(Gi \| \beta_i \| t_1 \| h(x_2 || x_1) \| Hi)$ is computed by user and the session key $s_k^* = h(Gi^* \| \beta_i \| t_1 \| h(x_2 || x_1) \| Hi^*)$ is computed by server in the last of each session. Hence, both of user and server can exchange their information securely and user will access the desired services from server safely because this scheme also provides forward secrecy, that is why our scheme must provide reasonable along with safe session key.

**PERFORMANCE ANALYSIS**

Here we measure and evaluate various performance parameters of the proposed SPAS scheme i.e. the storage capacity, communication cost, computational cost and security parameters under various known attacks in contrast to other schemes (Kumari et al., 2014; Kaul & Awasthi, 2016; Chaudhary et al., 2015; Jung et al., 2016). Suppose time complexity of hash function is $t_h$ moreover time complexity of XOR function is $t_0$. In our scheme, we made some assumptions on parameters like random numbers, secret numbers, identity, password, and time-stamps i.e. 128-bits. The efficiency comparisons regarding various schemes are described in Table 2. Aforementioned table analyzes and computes storage cost, communication cost along with computational complexity cost over various schemes with our proposed scheme and sum-up in the last.
Table 2

Efficiency Comparison related with Memory requirements (in bits), Transmission cost (in bits) and Computational complexity cost (in bits)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory Space in smart card (bits)</td>
<td>6*128=768 bits</td>
<td>6*128=768 bits</td>
<td>5*128=640 bits</td>
<td>7*128=896 bits</td>
<td>4*128=512 bits</td>
</tr>
<tr>
<td>Transmission Cost (bits)</td>
<td>8*128=1024 bits</td>
<td>7*128=896 bits</td>
<td>6*128=768 bits</td>
<td>8*128=1024 bits</td>
<td>9*128=1152 bits</td>
</tr>
<tr>
<td>Computational Complexity Cost</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Registration Phase (User Side)</td>
<td>2t₀+2t₀</td>
<td>1t₀+2t₀</td>
<td>2t₀+2t₀</td>
<td>1t₀+2t₀</td>
<td>1t₀</td>
</tr>
<tr>
<td>Registration Phase (Server Side)</td>
<td>6t₀+4t₀</td>
<td>4t₀+3t₀</td>
<td>4t₀+3t₀</td>
<td>3t₀+3t₀</td>
<td>3t₀+1t₀</td>
</tr>
<tr>
<td>Login Phase</td>
<td>12t₀+11t₀</td>
<td>8t₀+10t₀</td>
<td>8t₀+12t₀</td>
<td>8t₀+9t₀</td>
<td>4t₀+3t₀</td>
</tr>
<tr>
<td>Authentication Phase</td>
<td>8t₀+4t₀</td>
<td>6t₀+3t₀</td>
<td>6t₀+9t₀</td>
<td>6t₀+4t₀</td>
<td>7t₀+4t₀</td>
</tr>
<tr>
<td>Password Change Phase</td>
<td>9t₀+10t₀</td>
<td>6t₀+7t₀</td>
<td>10t₀+12t₀</td>
<td>6t₀+7t₀</td>
<td>7t₀+3t₀</td>
</tr>
<tr>
<td>Sum of Computational Cost</td>
<td>37t₀+31t₀</td>
<td>25t₀+25t₀</td>
<td>30t₀+40t₀</td>
<td>24t₀+25t₀</td>
<td>22t₀+11t₀</td>
</tr>
</tbody>
</table>

The storage cost is defined as a number of parameters stored in the smart card. Parameters like \{Aᵢ, Cᵢ, Dᵢ, Eᵢ, Fᵢ, hash\} are saved in smart card’s memory in the scheme. Hence, cost of memory space/storage is 6*128=768 bits. Figure 2 shows the comparison graph of storage space estimation cost (in bits) of presented scheme SPAS along with various other relevant schemes.

![Memory Space in smart card (bits)](image_url)

Figure 2. Storage cost comparison
• The total number of bits used in transmitting in login and authentication stage for whole messages are termed as transmission or communication cost.

• The scheme uses 6 parameters in login phase as \{c_i, \beta_i, J_i, K_i, t_i, I_i\} which requires 6*128=768 bits and for mutual authentication the number of parameters are \{a, t_j\}, requiring 2*128=256 bits. Hence, overhead for communication becomes = 6*128+2*128 = 1024 bits. Figure 3 shows the comparison graph of communication cost estimation (in bits) of our scheme along with various other relevant schemes.

• At user end, SPAS uses two hash functions along with two XOR function during registration phase. Therefore 2t_h(.) + 2t_\oplus is the computational complexity. Correspondingly, the server uses six hash function along with four XOR function during the registration phase, therefore, computational complexity is 6t_h(.) + 4t_\oplus at server end.

• During login stage, the scheme needs twelve hash functions along with eleven XOR functions to produce login request. Hence, 12t_h(.) + 11t_\oplus is the computational cost for login message.

• During mutual verification phase, a user needs one hash function, as well as server, needs seven hash along with four XOR functions. Hence, total computational cost for verification/authentication requires 8t_h(.) + 4t_\oplus operations.

• Figure 4 shows the comparison graph of computational complexity estimation cost in (bits) of scheme versus other relevant schemes. In the suggested scheme, computational complexity cost in (bits) is slightly higher than other’s scheme (Kumari et al., 2014; Chaudhary et al., 2015; Jung et al., 2016) which is not
baseless as this increased computational cost prevents it from different attacks possible in the network.

• After analysis, we found that schemes as Kaul and Awasthi, (2016); Chaudhary et al. (2015) and Jung et al.(2016) also suffer various attacks like server pose violation, conspirator violation, user impersonation attack, off-line password attack.

• Since the scheme uses low memory (bits) and having low computational cost (bits), and communication cost (bits) as well as secure against various attacks, therefore, we concluded that suggested idea/scheme performed most excellent from others so that we can implement it more practically than others over insecure networks.

• We demonstrated the security analysis of various protocols in Table 3 and achievements/goals in Table 4. Here, Figure 5 shows graph of security characteristics comparison and Figure 6 presents the graph of goal/achievements comparison.

![Computational Complexity Cost (in bits)](image)

**Figure 4. Computational Cost Comparison**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Conspirator Attack</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Chip Card Loss attack</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Table 3*

Security characteristics of our suggested scheme along with various other relevant schemes
Table 3 (Continue)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>User mask violation/attack</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Server mask violation/attack</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>On-line password assumption violation</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Off-line password assumption violation</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Replay violation</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>Denial of service violation</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Man in middle attack</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Lifted verifier violation</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Figure 5. Security Characteristics Comparison Graph
Table 4

Goal/Achievements Comparison

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User’s anonymity and un-traceability</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Support forward secrecy</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>Maintain Mutual Verification</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Compromise secret key</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>Compromise session key</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>Single registration</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Freely change password</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>No need of verification table</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Provide fast incorrect password checking facility</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>Scope of verification characteristics in chip card</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>Storage, functional as well as transmission cost must be low</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

We implemented the scheme in Python and the security features are tested/validate on AVISPA (Automated Validation of Internet Security Protocols and Applications) tool. AVISPA is having four tools to check the validation of security protocols.

- On-the-fly Model-Checker (OFMC)
- Constraint-Logic-based Attack Searcher (CL-AtSe)
- SAT-based Model-Checker (SATMC)
- Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP)
Figure 7 shows process flow diagram of AVISPA tool. The security analysis of the presented scheme has confirmed its performance in terms of reliability and safety because all the observations, trials, measuring operations return the same outcome on repeated testing. Moreover, the scheme is free from outside attacks, hazard, and insecurity, threat arising from loss of smart card. Analysis of the scheme shows that it is robust against all known attacks and ensures anonymity and privacy as there is no adverse effect of any types of attacks and it can withstands all such types of rigorous conditions. Moreover, its expandability characteristic is having the ability to support extra network users.

Figure 6. Goal/Achievements Comparison Graph

![Graph showing comparison of goals and achievements]
CONCLUSION

The SPAS scheme is introduced in this paper. The scheme ensures security, privacy and confidentiality of a user. It is an improvement over all the existing schemes (Chaudhary et al., 2015; Das et al., 2004; Jung et al., 2016; Kaul & Awasthi, 2016; Kumari et al., 2014). During investigation we found that earlier works were not secure enough for practical applications because all security parameters can be easily obtained by the challenger and are vulnerable to chip card loss violation as well as user un-traceability violation attack. Moreover, an adversary can get server’s secret key, as well as password of the entire registered user’s and also the session key for server, which may lead to destroying the whole system. During performance analysis of the scheme it is found that it incurs some extra bits of memory, and increased computational and communication cost but it helps to prevent smart card loss and user anonymity violation attack. The analysis of the scheme has confirmed its feasibility and performance in practical approach. The proposed scheme may be used in such applications which providing privacy protection with low-computation-ability devices. Thus, our idea is practically more acceptable to operate secure remote access over the public environment as well as may be simply integrated into various types of services such as academics, banking, and business applications. After performance and efficiency comparison, we demonstrate that suggested idea is safer as well as relevant to practical approach.

REFERENCES


