# An Improved Smartcard Based Password Authentication with Enhanced Security

**D. Dhayalan*, K. Malathi, P. Kirubai Nesam, P. Prasannakumari**
Dept. of Computer Applications, Vel Tech High Tech Dr.Rangarajan Dr.Sakunthala Engineering College, Anna University, Chennai, India.
***Corresponding author: Email: dhayalan@velhightech.com**

**ABSTRACT**

We offer a solution for user crime and server covenant drawback. We meant a smartcard based authentication that will exactly choose the feasible potential of a rival and counseling extensive set of twelve properties mounted as a valuable technique. Password hacking and smart card attack are often done. Even if the passwords are keep firmly, it can be leaked from the user side. To conquer this issue, we determine a security model by accommodating "honey words", and the attention of system security with a "fuzzy-verifier". In Leakage-resilient password system, a user should give a password diffusely, it may be additional burden. LRPS should use the trusted devices to identify password outflow on user side to realize each security and usefulness. In this, we had projected five aspects of authentication to handle each smartcard misplace attack as well as password deduction attack.

**KEY WORDS:** Hash values, LRPS, Fuzzy-verifier.

## 1. INTRODUCTION

Password predicated authentication is widely used and acceptable method for user access as a result of its facile-operation and affordable. The password-only mechanism of an intrinsic obstruction is that the server has to store passwords of all users in sensitive booster table. The password will be unsealed once the authentication server is compromised, even if the password square measure opportunely keep within the salted-hash, due to following reasons: the distribution of utilize-culled passwords are extremely skew, password cracking hardware (eg:GPUs) and algorithms (egg: Markova-chain-predicated). Threshold password-only authentication schemes have been proposed for password leakage from compromised server. In this, the password files and utilize information square measure shared over multiple servers, and thus no fusion of servers up to a bound threshold can learn something regarding the password and they are inherently unable to deal with password leakages at the utilize side (egg: obnubilated camera). To surmount this quandary, LRPS has been advocated. Through the aspects and generated hash values it can be secured.

This scheme includes the following modules,

- Registration
- Login
- Verification
- Forget password
- Change password

**Contribution:**

**Break-Fix-Break-Fix Cycle:** In this method it is refers to fee-for-service to providing the knowledge technology service to the business. While victimization this methodology it offer the services might embody rehabilitation, installation of system components, peripheral equipment is non-essential device connected to a host laptop and extends its capabilities.

- In this methodic framework we appraise the two-factor authentication schemes. The efficiency of 2 issue authentication its offer further layer of security and used to management the access of sensitive system and knowledge and on-line services it's terribly tougher for the assaulter to access the Arcanum
- By accommodate "honey words" with our suggested "fuzzy-verifier", Our designed can timely expose the user card fraudulency to on-line conjecturing and well address the obviously intractable security and usability issue
- We utilize extraordinarily colossal-scale authentic-life Arcanum to read the adequacy of our integration of "honey words" and the "fuzzy-verifier". We further show that our integration technique is generic and cannot be applied to 2 -factor schemes for varied alternative environment.

**Leakage resilient password system:** In volumetric analysis proves that a secure LRP system is designed for skilled operating within the sure secure channels unconcerned. The secure channel protects the mappings between original password symbols and associated random symbols. Unlike previous LRP system users, Shadow Key users do not have to remember something except their passwords. Leakage resilient watchword system is designed for professionals operating within the security business.

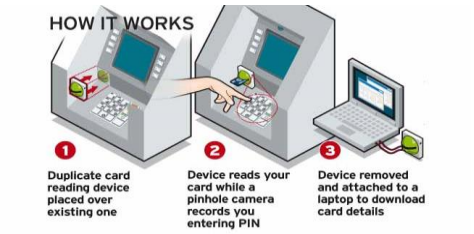


**Fig.1. Password leakage**

**Architecture:**

**Attacker model & Decision criteria:**

**Attacker's model:** In this we utilize protocol to authenticate a password for exchanging the keys and to design a system as a collection of running and interacting the threads of a project. It is a model used to control the communication set such as insertion, deletion, modification. And it will corrupt the person who is communicating in through this. Yet we are using another model of system that is, smart based security. Recently it has been proclaimated that, secret parameters that can be extracted in smart cards by power analyze attack. And the leakage of sensitive parameters results in the open-end credit loss drawback. Sometimes users might forget their passwords and use to place numerous recollected passwords that results in to password block.

**Decision criteria:**

**Table.1. The broad list of 12 separate decision criteria**

| | | | |
|---|---|---|---|
| 1 | Forward mystery | 7 | No password risk |
| 2 | Password friendly | 8 | No password verifier-table |
| 3 | Utilize anonymity | 9 | Provision of key accord |
| 4 | No smart card loss attack | 10 | Tough to know attacks. |
| 5 | Sound reparability | 11 | Sound reputability |
| 6 | No clock synchronism | 12 | Collective authentication |

**Hash Values:** This paper includes the hash values for securing the data. A hash worth is a numeric value of a fine-tuned length that unambiguously identifies knowledge. Hash values represent astronomically immense amounts of knowledge, you can sign a hash worth a lot of expeditiously than linguistic communication them a lot of vastly large worth. Hashing is the transformation of string of character into a customarily shorter fine-tuned-length worth or key that represents the pristine string. It is withal used in several coding algorithms.

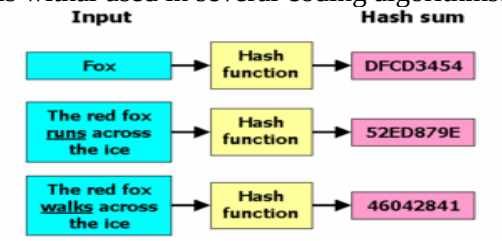


**Fig.2. conversion using hash functions**

Some relative simple hash functions have been used:

**Division method**: The size of the number of items in the table is prognosticated. That number is then utilized as a divisor into each pristine value or key to dissever a quotient and a remnant. The remnant is the hashed value.

**Folding method**: This method divides the pristine value into several components, integrate a component together and then utilize the last four digits as the hashed value or key.

**Source transformation method:** where the worth or secret is digital, the number base are often amendment leading to a distinct sequence of digit. High order digit could be discarded to suit a hash worth of uniform length

**Digit rearrangement method**: This is simply taking part of the original value or key such as digits in position 3 through 6 reversing there order, and then using that sequence of digits as the hash value or key.

## 2. MATERIALS AND METHODS

We gift associate economical smart-card predicated parole authentication theme that is able to offer all of the twelve criteria introduced in sec.2.2. It generate the hash values for both server and user facet to provide. Our scheme consists of four phases: registration, login, verification password change and forget password.

**Table.2. Notations and Abbrevations**

| Symbol | Description | Symbol | Description |
|---|---|---|---|
| $U_i$ | $i^{th}$user | $\oplus$ | The bitwise XOR operation |
| S | The bitwise XOR operation | $\|\|$ | String concatenantion |
| A | The adversary | $x$ | The secret key of S |
| $ID_i$ | Identity of user $U_i$ | $\longrightarrow$ | A common channel |
| $PW_i$ | Password of user $U_i$ | $\Longrightarrow$ | A secure channel |

**Registration:** The registration performs as follows:

Step 1. Ui Chooses IDi, PWi and random string b.

Step 2. Ui= S :{ IDi, HO (b||P2Wi)}

Step 3. After receiving the registration message at a time t, S computes Ai = H0 ((H0 (IDi) ⊕ H0 (b‖PWi)) modn0), If the user is new, it will enter into the new entry and saves {IDi, Treg=T, ai, Honey_List=NULL} in this entry or S only updates the values of Treg to T, ai to newly created ai, and Honey_Listto NULL in the existing entry for Ui.

Step 4. S computes Ni = H0 (b‖PWi) ⊕ H0 (x‖IDi‖Treg).

**Login:** This involves the following operations:

Step 1. Ui inserts her smart card SC into the reader and inputs ID∗i, PW∗i. [1]

Step 2. Smart card generates A∗i = H0 ((H0 (ID∗i) ⊕ H0 (b‖PW∗i)) mod n0) and verifies the IDi and PWi that equals the Ai value of stored ai [2].If they are not equal it will terminated.

Step 3. SC chooses u and computes C1 = gu mod p, Y1 = yu mod p, k = H0 (x‖IDi‖Treg) = Ni ⊕ H0 (b‖PW∗i), ai= (Ai ⊕ai) ⊕Ai, CIDi = ID∗i⊕ H0 (C1‖Y1), CAKi = (ai‖k) ⊕H0 (Y1‖C1) and Mi = H0 (Y1‖k‖CIDi‖CAKi).

Step 4. Ui sends to server S: {C1, CIDi, CAKi, Mi}.

**Verification:** After the login process, it will proceed the verification steps:

Step 1. S computes Y1 = (C1) x mod p using private keyx and derives IDi = CIDi⊕H0 (C1‖Y1) andchecks whether IDi is in the correct format. If it is not valid,it will be terminated.

Step 2. S computes k=H0 (x‖IDi‖Treg) and M∗i =H0 (Y1‖k‖CIDi‖CAKi).

Step 3. S derives ai||k = CAKi⊕H0 (Y1‖C1), and checks Whethera′i equals the stored ai.

Step 4. S generates a random number v and computes the temporary key KS = (C1) v mod p, C2 = gv mod p and C3 = H1 (IDi‖IDS‖Y1‖C2‖k‖KS).

Step 5. S → Ui: {C2, C3}.

Step 6. On receiving the reply message from the server S, the smart card computes KU = (C2) u mod p, C∗3=H1 (IDi‖IDS‖Y1‖C2‖k‖KU), and comparesC∗3 with the received C3, C4 = H2 (IDi‖IDS‖Y1‖C2‖k‖KU).

Step 7. After receiving {C4} from Ui, S first computes C∗4 = H2 (IDi‖IDS‖Y1‖C2‖k‖KS) and then checksif C∗4 equals the received C4. If this verification holds, S authenticates Ui and the login request is accepted. Otherwise, the connection is terminated.

Step V9. The user Ui and the server S agree on the commonsession key skU = H3 (IDi‖IDS‖Y1‖C2‖k‖KU) = H3 (IDi‖IDS‖Y1‖C2‖k‖KS) = skS for securing future data communications.

**Password change**: This performed locally without the hassle of interaction with the remote server, and it involves the following steps:

Step 1. Ui inserts her smart card into the card reader and inputsIDi and the original password PWi.

Step 2. The card computes A∗i = H0 ((H0(IDi) ⊕ H0(b‖PWi)) mod n0) and verifies the validity of A∗iby checking whether A∗i equals to the stored Ai. If the verification holds, it implies the input Idi and PWi are valid with a probability of $n0-1n_0$ (≈ 99:61100, when n0=28). Otherwise, the smart card rejects.

Step 3. The smart card asks Ui to resubmit a new password PWnewi and computes Nnewi = Ni ⊕H0 (b‖ PWi) ⊕H0 (b‖PWnewi), Anewi = H0 ((H0 (IDi) ⊕H0 (b‖PWnewi) mod n0). Then, smart card updates the values of Ni, Ai and ai⊕Ai with Nnewi , Anew I and ai⊕Anewi, respectively.

**Forget Password**: Ui inserts her smart card into the card reader and inputs IDi. The card computes A∗ i = H0 ((H0(IDi) and verifies the validity of A∗ i by checking whether A∗ i equals to the stored Ai. If the verification holds, it implies the input IDi and PWi are valid with a probability of n0−1 n0 (≈ 99:61 100, when n0=28). Otherwise, the smart card rejects. Step P3 if true the data will be provided to the user

**Effectiveness of "fuzzy-verifier" + "honey words":** In this scheme, to provide the admired property of "local and secure countersign change", Uistores the "fuzzy-verifier"Ai=H0((H0(IDi)⊕H0(b‖PWi)) mod n0) in its card memory. This also facilitates A to cut back her countersign emphasize that, the values for m0and n0can be adjusted to cater for diversified security demands in totally different systems.

Due to highly inclined countersign distribution in point of fact, the conventional optimum security sure for the most part underestimates A's benefits and engenders a false sense of security. Our scheme achieves security (Pr [Succ Ext] = C′• ms′0≈Σm0j=1pi ≤Σ10j=1pi) on the far side the typical optimum security sure serving as a hedge against human-being' limited memory.

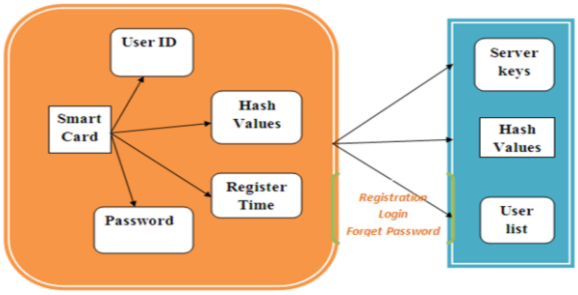


**Figure.3. Proposed system Architecture**

## 3. RESULTS AND DISCUSSION

**Formal security analysis:** Here we use two theorems as followed,

**Theorem1**: Let G be a representative group, D be a password area with its frequency distribution following the Zipf's law [4], and n0 be the "security-usability tradeoff parameter". Let Pbe the proposed theme expressed. Let A be a PPT adversary against the semantic security inside a time certain t, with q send (≤ m0 = 10) Send-queries and qexe Execution-queries, and making less than qh random oracle queries. Then we have

$$\mathrm{Adv}^{\mathrm{ake}}_{P;D}(A) = 2\Pr[\mathrm{Succ}_7] - 1 + 2(\Pr[\mathrm{Succ}_0] - \Pr[\mathrm{Succ}_7])$$
$$\leq C' \cdot q'^{s}_{send} + 12qh\mathrm{Adv}^{\mathrm{CDH}}_{P}(t + (qsend + qexe + 1) \cdot \tau e) +$$
$$((q^2h + 8qsend) / 2l) + ((qsend + qexe)2 / p)$$

Where we tend to use the Zipf model of the Ti any a password distribution in, where |D|=12,898,437, C′=0.062239 ands′=0.155478; n0 = 28; τe is the computation time for an exponentiation in G, and l = min, i = 0, 1, 2, 3. Proof. Let A be an human against the linguistics security of our theme. Our main idea is to use A to construct probabilistic polynomial-time (PPT) adversaries for every of the underlying primitives (e.g. Hash and CDH intractability) in such a way that if A manages to interrupt the semantic security, then at least one in all these PPT adversaries succeeds in breaking the protection of an underlying primitive. We prove Theorem 1 through a series of hybrid games Gn (n =0, 1. . . 8), starting with the real attack G0 and ending in G8where A's advantage is zero, and for which we will sure the difference in A's advantage between any 2 consecutive games. The detailed proof will be found in Appendix C-1.

**Theorem 2**: G, D, n0 and P are of the same meaning with those of Theorem one. Let A be an someone against mutual authentication within a time sure t, with less than q send (≤ m0 = 10) Send-queries and qexe Execution-queries, and making less than qh random oracle queries. Then,

$$\mathrm{Adv}^{\mathrm{auth}}_{P;D}(A) \leq C' \cdot qs'send + (q2h + 8qsend) / 2l+1) +$$
$$((qsend + qexe)2) / 2p)$$
$$+5qh\mathrm{Adv}^{\mathrm{CDH}}_{P}(t + (q_{send} + q_{exe} + 1) \cdot \tau_e),$$

In this section, we compare the performance and the fulfillment of the criteria among relevant schemes and our proposed scheme. The comparison results are depicted in table.3.

**Table.3. Performance Comparison among Relevant Authentication Schemes**

| | | Computation overhead | | Communication Cost | | |
|---|---|---|---|---|---|---|
| | Protocol rounds | User side | Server side | User Side | Server side | Storage cost |
| Xu (2009) | 2 | 3TE+5TH≈ 481.305ms | 3TE+4TH≈ 8.615 ms | 1408 bits | 1408 bits | 3200 bits |
| Wang (2012) | 3 | 3TE+7TH≈ 505.827ms | 3TE+5TH≈ 8.616 ms | 1408 bits | 1152 bits | 3456 bits |
| Wu (2012) | 3 | 4TE+5TH≈ 621.305ms | 3TE+5TH≈ 8.616 ms | 2304 bits | 1152 bits | 3456 bits |
| Li (2013) | 2 | 4TE+4TH≈ 609.044ms | 3TE+3TH≈ 8.615 ms | 1408 bits | 1408 bits | 4096 bits |
| Jiang (2015) | 2 | 4TE+4TH≈ 609.044ms | 2TE+4TH≈ 5.744 ms | 2304 bits | 1152 bits | 3328 bits |
| Truong (2015) | 3 | TC+7TH≈ 525.327 ms | 3TC+7TH≈30.775 ms | 640 bits | 1152 bits | 384 bits |
| Islam (2016) | 2 | 3TE+3TH≈ 456.783ms | 2TE+3TH≈ 5.744 ms | 1408 bits | 1408 bits | 2176 bits |
| Our scheme | 3 | 3TE+9TH≈530.349 ms | 3TE+7TH≈ 8.617 ms | 1536 bits | 1152 bits | 3616 bits |

Without loss of abstraction, the security framework n0 is assumed to be 32bit long, the identity IDi, password PWi, random numbers, timestamps and outcome of hash functions are all counseled to be 128 bit long, while y and g ar 1024 bit long.

Let, TH = Time complexity for hash, TE = Modular mathematical process, TS= Symmetric secret writing, and TC=Chebysev polynomial

To have a more intuitive grasp on the computation overhead of our theme, in Table.3, we have a tendency to list the computation time for connected crypto graphical operations on totally different platforms. We use a Philips HiPerSmartTM card to approximate user device, and the computation time of related operations is according. This smart card is provided with a 32-bit reduced instruction set computing MIPS-based processor, offering a most clock

speed of thirty six rate, as well as a 2 computer memory unit instruction cache, 256 computer memory unit flash memory and sixteen KB RAM. We use common Laptops to approximate the server and judge the server facet computation price. Note that both our implementation and that of create use of the quality crypto graphical library MIRACL, which is a multi-precision integer and rational arithmetic C/C++ library.

**Table.4. Timings for cryptographic operations**

| Experimental platform | Exp.$T_E$ (\|p\|=1024) | Chebysev $T_C$ (\|p\|=1024) | Symm.$T_S$ (AES-128) | Hash $T_H$ (SHA-1) |
|---|---|---|---|---|
| Philips HiPerSmart 36 MHz | 140.0ms | 439.5ms | 4.972ms | 12.261ms |
| Intel(R) T5870 2.00 GHz | 10.257ms | 32.200ms | 2.012µs | 2.580µs |
| Intel(R) i7-4790 3.60GHz | 2.871ms | 10.257ms | 0.086µs | 0.598µs |

As shown in Table.3, our scheme provides all the twelve criteria while maintaining reasonable efficiency; all the other schemes fail to achieve at least one critical criterion because of security pitfalls or due to a violation of the inherent security-usability conflict revealed, they only employ conventional cryptographic approaches.

## 4. CONCLUSION

In this paper, we have taken a first step towards breaking the "break-fix-break-fix cycle" in the two-factor authentication research area. In advance of our proposal of a new scheme which meets contingency, primitiveness, and strong notions of security, the proposed adversary model and criteria set provide a benchmark for the evaluation of current and future two-factor authentication proposals. To the best of our knowledge, we, for the first time, introduce "honey words", traditionally the purview of system security, into two-factor cryptographic protocol design. By integrating "honey words" with the proposed "fuzzy-verifier", our scheme can timely detect user card corruption to thwart online guessing and well addresses the seemingly intractable security-usability issue left. Particularly, eleven large scale password datasets, which consist of 102.6 million real-life passwords and cover various popular services and diversified user bases (e.g., language), are used to establish the practicality of the proposed approach.

## 5. ACKNOWLEDGEMENT

The author wish to thank Val Shree Dr. R. Rangarajan, Chancellor, Val Tech High Tech Dr. RR and Dr. SR Engineering College, for the support and facilities provided for the preparation of this paper.
**Financial disclosure:** No financial support was received for this implementation.

## REFERENCES

Huang X, Chen X, Li J, and Xiang L, Yang Xu, Further observations on smart-card-based password-authenticated key agreement in distributed systems, IEEE Trans. Para. Distrib. Syst., 25 (7), 2014, 1767–1775.

Juels A, and Rivest R.L, Honey words: Making password-cracking Detectable, in Proc. ACM CCS, 2013, 145–160.

Scott M, Costigan N, and Abdulwahab W, Implementing cryptographic pairings on smartcards, in CHES, 2006, 134–147.

Wang D, D. He D, P. Wang P, and C.-H. Chu C.H, Anonymous two-factor Authentication in distributed systems: Certain goals are beyond Attainment, IEEE Trans. Depend. Secur. Comput, 12 (4), 2015, 428–442.

Wu S.H, Zhu Y.F, and Pu Q, Robust smart-cards-based user Authentication scheme with user anonymity, Secur. Commun. Netw., 5 (2), 2012, 236–248.