Int J Syst Assur Eng Manag (June 2022) 13(3):1135–1150
https://doi.org/10.1007/s13198-021-01408-0

ORIGINAL ARTICLE

Enhanced and secured random number generation for eUASBP

Sangeetha Rajaram[1] · **Satyanarayana Vollala**[2] · **N. Ramasubramanian**[1] · **J. Kokila**[3]

Received: 15 May 2021 / Revised: 14 July 2021 / Accepted: 15 September 2021 / Published online: 17 November 2021

Abstract Authentication is needed for vital applications around the world. Applications that deal with sensitive information need to be kept secret through anyone type of authentication mode. Password-based smart card authentication is a generic model of authentication. Many password-based smart card authentication protocols have been proposed by Sangeetha et al., Giri et al., Jia et al. , Das et al. and so on. Those protocols provide protection against many type of attacks such as offline password guessing attack, server masquerading attack, user impersonation attack, privileged insider attack, but they are open to Crypt analytic attack, Cloning attack, and Reverse engineering attack. This paper proposes an authentication protocol that is termed as *hrngRO*- hybrid RNG by RO-PUF and SHA-256 for *eUASBP* simulated using AVISPA tool. It provides Ring Oscillator PUF based random number generator using the SHA-256 hash function. This hybrid combination of RNG makes *hrngRO* more secure against Crypt analytic attack, Cloning attack, Reverse engineering attack and makes the system even faster. Pre Early Wrong Password Detection is supported by *hrngRO* using bloom filters. *hrngRO* provides protection against many types of attacks like other authentication protocols and it also provides additional security by avoiding the above-mentioned attacks. Security of *hrngRO* is tested using AVISPA tool.

Keywords AVISPA · PUF · RNG · Bilinear pairing

1 Introduction

Present lifestyle and COVID-19 (Coronavirus disease 2019) pandemic spread encourages online services instead of traditional services that need human interaction. Authentication is needed for essential online services that deal with money flow and private information. Remote servers need to authenticate users to allow the authorized ones and to protect them from unauthorized ones. Sensitive applications that type deal with money flow and private information attract many kinds of adversaries and attacks towards money theft and information theft like ants attracted towards sugar.

Three common modes of authentication are smart cards, passwords, and biometric measures. Most of the present applications make use of smart cards and password combinations for security. Upcoming applications and few existing applications make use of biometric measures for security. This paper deals with password-based smart card applications and some of the password-based smart card applications are:

- Vehicle or people monitors: Tracking and monitoring people or vehicles in a particular region that comes under wireless sensor network field.
- Attendance calculation: Attendance calculation of persons in educational institutions or commercial organizations is done by smart cards.

✉ Sangeetha Rajaram
406913002@nitt.edu

[1] Department of Computer Science and Engineering, National Institute of Technology, Tiruchirappalli 620015, India

[2] Department of Computer Science and Engineering, IIIT, Naya Raipur, India

[3] Department of Information Technology, Indian Institute of Information Technology, Allahabad, India

- Metro station enhancement: Smart cards have been used to estimate the waiting and walking time of passengers in metro stations to benefit elders and physically disabled passengers Li et al. (2020).
- Banking: ATM (Automatic Teller Machine) cards perform most of the banking operations like deposit, withdrawal, PIN (Personal Identification Number) change, money transfer, and so on. Recent ATM cards make use of biometric measures too, to ensure additional security.
- Purchase Cards: Most of the malls and shops make use of purchase cards to value the customers and to provide gifts and offers based on them.
- Travel agencies: Smart card data are being used to track the employment status of the passengers to schedule travel according to their working days or holidays Zhang and Cheng (2019).
- Traffic Management: Nowadays, smart cards have been used to calculate traffic flow in the subways during special events Chen et al. (2019).

Password-based authentication is a type of authentication used by applications commonly. Most of the applications allow the users to select a password and some of the applications generate a password for the users. Some of the norms to be followed while selecting a password are:

- Password selected should not be easily guessable.
- Password must restrict offline and online password guessing attacks.
- Password has to be changed frequently to provide better protection.
- New password may be different from the old one and unrelated.
- Password should be kept secret and it should not be revealed to anyone or the public.
- Password should not contain family details or any other publicly known details like mail id or phone number.
- Users must hide password's exact time of use to access a particular service He et al. (2020) Sun et al. (2011) Furnell (2020) Raponi and Di Pietro (2020).

Abbreviations used in this paper are listed in Table 1.

Most of the sensitive applications like banking use EMV smart cards for its unclonable property Bond et al. (2015) El Madhoun et al. (2018). Therefore smart cards must be kept safe from intruders and adversaries. *eUASBP* provides ECC based authentication protocol that reduces the computational cost. ECC replaced public key cryptography with its minimal key size without sacrificing security. ECC-based authentication is used by most IoT applications for its compactness towards keysize Lara-Nino et al. (2020). It is suitable for lightweight applications. *eUASBP* provides less computation cost, early wrong password detection, mutual authentication, and protection from most of the attacks.

Some of the terminologies used in this paper are discussed below.

- Bloom Filter: Bloom filter is a data structure used to ensure the membership of an element in a set. It stores 1 or 0 in the calculated bit indices using any of the hash functions. For the presence of a particular value, 1 is stored and for the absence of a particular value, 0 is stored. Bloom filters support only false positives and never support false negatives Sangeetha and Ramasubramanian (2015).
- RO-PUF: RO-PUF is a delay-based electronic PUF. RO-PUFs are formed by connecting the odd number of inverters in a feedback loop that oscillates the input signal. The number of oscillations per time unit differs for every implementation of RO-PUF McGrath et al. (2019) Kokila et al. (2019) Kokila and Ramasubramanian (2019).
- SHA-256: Input of this algorithm is of variable size. In the preprocessing or padder unit, variable-sized inputs are padded to make it a fixed size for processing. Message blocks are segmented and padded as required. Message blocks are scheduled in the scheduler unit. In the digest or compression unit, repeated compression for all the message blocks is being done and updations are done from temporary variables into the final hashed output variable. Control unit controls all the other units Cao and O'Neill (2012) Thakur and Malviya (2018). SHA-256 algorithm generates a fixed-size output on variable-size inputs. SHA - 256 algorithm is reliable because small changes in input change the output.
- RNG: Blocks or modules specifically used for random number generation. Used as a part of any complex or purposeful system.
- PRNG: These are based on calculations and software. So, PRNGs are easily reproducible.
- TRNG: These are based on the physical phenomenon and difficult to replicate.

Strong RNGs improve the performance and security features of authentication protocols. TRNGs are based on physical entropy. PRNGs are based on manipulations and software. HRNGs improve security features in a much better way than the other two mentioned. This paper proposes hybrid RNG using RO PUF and SHA-256 for *eUASBP* with AVISPA simulations called *hrngRO*. *hrngRO* supports Pre Early Wrong Password Detection and provides protection against attacks such as Crypt analytic attack, Cloning attack, and Reverse engineering attack.

- Pre Early Wrong Password Detection: Wrong entry of username and password are detected by the use of

Table 1 Abbreviations Used

Abbreviation	Expansion
EMV	Europay, Mastercard, and Visa
eUASBP	enhanced User Authentication Scheme based on Bilinear Pairing
ECC	Elliptic Curve Cryptography
IOT	The Internet of Things
RNG	Random Number Generator
HRNG	Hybrid Random Number Generator
TRNG	True Random Number Generator
PRNG	Pseudo Random Number Generator
PUF	Physically Unclonable Function
RO-PUF	Ring Oscillator PUF
SHA-256	Secure Hash Algorithm FIPS 182-2
AVISPA	Automated Validation of Internet Security Protocols and Applications
RTL	Register Transfer Level
HLPSL	High-Level Protocol Specification Language

bloom filter, before proceeding with the steps of authentication protocol. Pre Early Wrong Password Detection avoids the execution of authentication protocol on wrong entry of username or password, thereby saves the remote server's execution time.

- Crypt Analytic Attack: *hrngRO* is a combination of SHA-256 algorithm and RO-PUF. SHA-256 algorithm and RO-PUF act as one way hash functions. This double inversion property makes Crypt Analytic attack impossible with *hrngRO*.
- Cloning Attack: It becomes impossible with *hrngRO* because of the physical entropy property of RO-PUF and inversion property of SHA-256 algorithm.
- Reverse Engineering Attack: It becomes impossible with *hrngRO* because of the physical entropy property of RO-PUF.

It provides enhancements over *eUASBP* with the following modules Rajaram et al. (2019),

- Random number generator module: Uses SHA-256 hash function and RO-PUF. Hashed form of the user entered password using SHA-256 hash function is fed as a input to RO-PUF to generate random number for registration and authentication phases of *hrngRO*.
- Preliminary user credential checking module: Uses bloomfilter for performing this operation. Hamming distance between random numbers of the userid and password are used to generate bit indices of the bloom filter.
- Security module: Does formal and informal security analysis of *hrngRO*. Formal security analysis is done using AVISPA tool. *hrngRO* provides protection against the following attacks, Crypt analytic attack, Cloning attack, Reverse engineering attack, Offline password guessing attack, Session key disclosure attack, Insider attack, User impersonation attack, Theft attack, Smart card stolen attack, Server masquerading attack.
- It provides Pre Early Wrong Password Detection, Mutual authentication, and Session key agreement.

Most of the authentication protocols use PUFs as one way hash function, key generator, and lightweight module based on the application but *hrngRO* uses it for secure random number generation. Feng Zhu et al. proposed an authentication protocol that uses arbiter PUFs to generate tags Zhu et al. (2019). Lee, Tian-Fu et al. also proposed an lightweight authentication scheme using Barrel Shifter Physically Unclonable Function (BS-PUF) for IoMT to secure the sensors and fog nodes and to evade a computational load on devices Lee and Chen (2021) RO-PUF and SHA-256 hash function are proven to be secure enough in the applications they have been used individually. A combination of RO-PUF and SHA-256 hash function will also be more secure than their separate use. Hence this combination is resisted over Crypt analytic, Cloning, and Reverse engineering attack.

Rest of the paper is organized as follows. Section 2 provides related works. Section 3 explains Working of *hrngRO*. Section 4 deals with Implementation and Results of *hrngRO*. Section 5 provides formal and informal security analysis of *hrngRO*. Section 6 concludes the paper.

2 Related works

Authentication is needed by most of the online and offline services provided throughout the world. Smart card-based authentication depends on the physical presence of the smart card. Password-based authentication is simple enough and supported by many applications. Smart card-based password authentication is considered in this work. This section discusses the competent Smart card based password authentication protocols.

Salman et al. reviewed a protocol based on bilinear pairing for IOT proposed by Nikravan-Reza. Their protocol is vulnerable to node impersonation attack, user impersonation attack, and other security attacks. So their protocol is not worth for practical implementation Shamshad et al. (2020). Guangsong Li et al. proposed a Priority Aware Anonymous Handover Authentication Protocol that supports user priority, user revocation, attribute revocation, and works based on attribute based cryptography Li et al. (2020). However it supports only two attributes. Due to its simple access structure it opens door for adversaries unknowingly. Jiangheng Kou et al. proposed a hierarchical authentication protocol based on Merkle tree that supports authentication right for multi-server architecture Kou et al. (2020). Here practical implementation of the protocol and testing are not being done by the authors.

Fang et al. proposed an improved authentication scheme by analyzing some of the contemporary authentication schemes to overcome their limitations. Even then it is prone to offline attacks Fang and huang (2006), Awasthi (2012). Giri et al. provide an improved version of the authentication scheme that overcomes the flaws of some contemporary authentication schemes. But it takes more computation time to execute its phases and it is prone to theft attack Giri and Srivastava (2006), Awasthi (2012).

Awasthi proposed an authentication scheme that overcomes the limitations of Fang et al. 's scheme and Giri et al. 's scheme that provides strong security over other schemes. Despite it is open to offline secure password change protocol problem Awasthi (2012). Bayat et al. proposed password-based authentication scheme that doesn't support early wrong password detection, and session key agreement Bayat et al. (2010), Rajaram et al. (2019). Manik et al. proposed a bilinear pairing based authentication scheme that allows local password changing provision. On the other hand, it is prone to most of the possible attacks Das et al. (2006), Rajaram et al. (2019). Chou et al. proved Manik et al. 's scheme is not secure enough to handle adversaries and vulnerable to user impersonation attacks. Chou et al. gave a solution to avoid that attack Chou et al. (2005).

Jia et al. proposed a remote user authentication scheme based on bilinear pairing and elliptic curve ElGamal encryption and decryption scheme. The proposed scheme is prone to most of the smartcard-based attacks Jia et al. (2006), Rajaram et al. (2019). Tseng et al. proposed a pairing-based user authentication scheme for wireless clients. Though it is suitable for lightweight devices, it is vulnerable to cloning and cryptanalytic attacks Tseng et al. (2008). Goriparthi et al. proposed an authentication scheme that overcomes the limitations of Chou et al. 's scheme and Das et al. 's scheme. Goriparthi et al. (2009). Goriparthi et al. 's scheme open the way for adversaries to perform attacks like cryptanalytic and cloning attacks. Hakeem et al.implemented and analyzed the proposed key generation and authentication methods using an authentication simulator and a bilinear pairing library Hakeem and Kim (2021).

Sangeetha Rajaram et al. proposed a protocol called *eUASBP* that works based on bilinear pairing. *eUASBP* proved to be secure against most of the smartcard-based attacks like server masquerading attack, theft attack, privileged insider attack, session key discloser attack, and so on. Additional advantages of *eUASBP* are session key agreement, mutual authentication, efficient password change phase, early wrong password detection. *eUASBP* proved to be secure formerly by using BAN logic Rajaram et al. (2019).

RNG module of any security system supports security directly or indirectly. An efficient RNG module ensures effective security of the system. Software-based RNG module supports Pseudo random number generation. PRNG is flexible for further developments. Hardware-based RNG module supports the generation of true random numbers. This paper combines SHA-256 algorithm and RO-PUF for the random number generation. This hybrid combination to generates random numbers avoids many attacks and ensures security.

Proposed *hrngRO* Protocol provides a random number generator module, Preliminary user credential checking module, and a strong security analysis module. *hrngRO* provides protection against the following attacks, Cryptanalytic attack, Cloning attack, Reverse engineering attack, Offline password guessing attack, Session key disclosure attack, Insider attack, User impersonation attack, Theft attack, Smart card stolen attack, Server masquerading attack. And it provides Pre Early Wrong Password Detection, Mutual authentication, Session key agreement.

3 Proposed *hrngRO*

Authentication is a crucial part of this digital era. Smartcard based authentication is being adopted by many applications like attendance maintenance of organizations or educational institutions, to perform banking operations using credit or debit cards, customer maintenance of shops to keep track of the shopping records etc., Many authentication protocols have been proposed by researchers for smartcard-based applications to protect the system against adversaries. *eUASBP* proposed by Sangeetha Rajaram et al. defends many attacks like insider attack, user impersonation attack, smart card stolen attack, and, so on. *eUASBP* has specific advantages over other authentication protocols like improved performance, session key agreement, early wrong password detection, and so on Rajaram et al. (2019). Registration and authentication phases of *eUASBP* need a random number generator for generating random numbers Rajaram et al. (2019). This paper proposes a Ring Oscillator PUF based random number generator for *eUASBP*using SHA-256 hash function called as *hrngRO*, ensures its security by using AVISPA tool and does preliminary userid, password checking using bloom filters. *hrngRO* combines SHA-256 hash function and RO-PUF for random number generation. The working of hrngRO is depicted in Fig. 1. hrngRO contains three different modules for different purposes on top of eUASBP. They are,

1. Security module: Authentication code is written in HLPSL and tested using AVISPA tool.
2. Random Number Generator module: Used to generate random numbers in registration and authentication phases of eUASBP. The depiction of the random number generator module is given in Fig. 2. On the entry of variable size input, SHA 256 hash function generates a fixed size 256-bit challenge. RO-PUF module generates a 128-bit random number by receiving 256-bit challenge.
3. Bloom Filter module: Finds hamming distance between random numbers of userid and password. Generates bit indices using the hamming distance calculated. Entries in the 128 - bit bloom filter is done using the bit indices calculated.

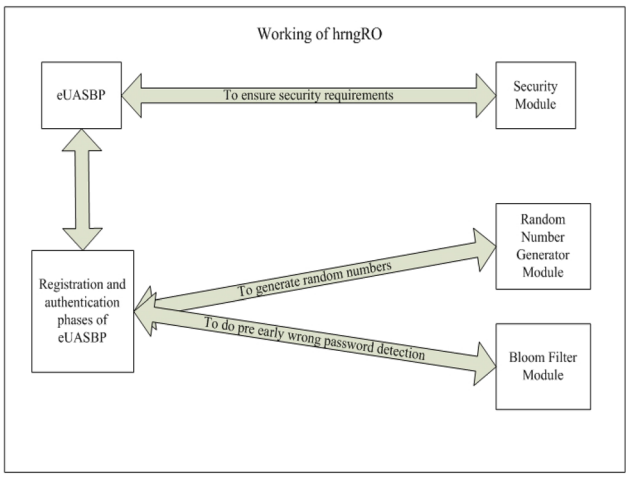

Fig. 1 Working of *hrngRO*

Depiction of random number generator module is given in the Fig. 2.

3.1 Working of *eUASBP* Rajaram et al. (2019)

eUASBP involves five phases as followed, Initialization phase, User's registration phase, User's login phase, User's authentication phase, and Password change phase Rajaram et al. (2019).

3.1.1 Initialization phase

Chooses cyclic groups G_{ro1}, G_{ro2}, bilinear map $\widehat{e} : G_{ro1} x G_{ro2}$, random generator $RP1 \in G_{ro1}$, cryptographic hash function $hfu_1 : \{0,1\}^* \rightarrow Z_q^*$ and a secret key sK. Computes $P_{pb} = sK.RP1$, bilinear map $\widehat{e}(RP1, RP1) = \propto$ and publishes all parameters as public except sK.

3.1.2 User's registration phase

User sends userid IDe_x and a hashed password $PWDe_x = hfu_1(PWe_x, ye)$ - Uses a random cardinal ye. Remote Server computes $Re_x = hfu_1(sK, IDe_x) \oplus PWDe_x$, $REGe_{IDx} = hfu_1(sK, IDe_x) \cdot PWDe_x \cdot RP1$, and stores $< Re_x, REGe_{IDx}, hfu_1(.), RP1, P_{pb} >$ in the smart card. User stores a random cardinal ye in the smartcard.

3.1.3 User's login phase

On the entry of IDe_x and PWe_x, card reader computes $PWDe_x^* = hfu_1 \quad (PWe_x \cdot ye)$, $Le_0 = Re_x \oplus PWDe_x = hfu_1(sK, IDe_x)$ and, $REGe_{IDx}^* = Le_0 . PWDe_x^* .RP1$. Matches $REGe_{IDx}^*$ and $REGe_{IDx}$. On successful match, card reader computes $Le_1 = hfu_1(ye_1, T_{xe})$, $Le_2 = hfu_1(PWDe_x^*, Le_1)$, $Le_3 = (Le_0 + Le_2) \cdot P_{pb}$, $Le_4 = PWDe_x^* \oplus Le_1$ on selecting a random cardinal ye_1 and sends $< IDe_x, Re_x, Le_3, Le_4, T_x >$ to the RS.

3.1.4 User's authentication phase

RS checks the format of IDe_x and the value of ΔT_e is less than $(T_{se} - T_{ue})$. Computes $Ae_0 = hfu_1(sK, IDe_x)$, $PWDe_x = Re_x \oplus Ae_0$, $Le_1{}^* = hfu_1(ye_1, T_x e)$, $Le_2{}^* = hfu_1(PWDe_x{}^*, Le_1{}^*)$. On successsful matching of $\widehat{e}(Le_{3x}, RP1)$ and $\propto^{(sK.(Ae_0 + Le_2))}$, RS computes $Ae_1 = hfu_1(ye_2, T_{se})$,

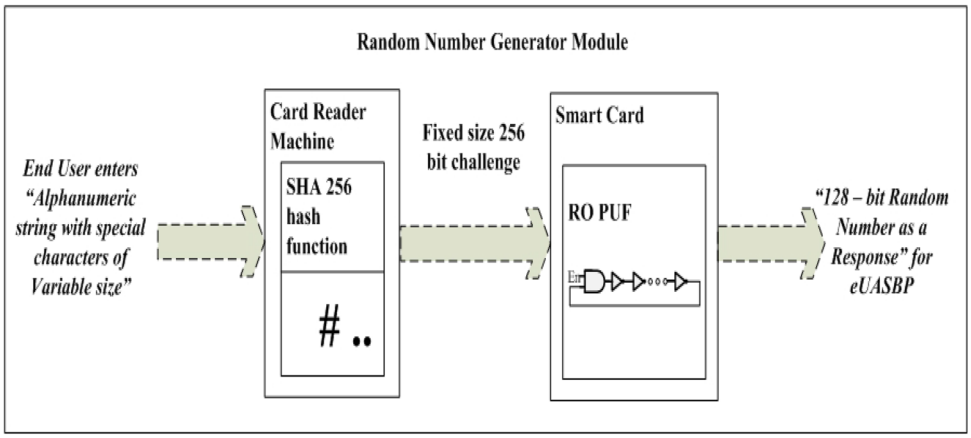

Fig. 2 Random number generator module

$Ae_2 = hfu_1(Ae_0, Ae_1)$, $Ae_3 = Ae_2.PWDe_x^*.RP1$, $ye_3 = Le_1^* \oplus Ae_1$, $ssKe = hfu_1(Ae_2, Le_1^*, Ae_1, Le_2^*)$ by choosing a random cardinal ye_2. *RS* sends $< Ae_3, ye_3, T_s >$ to the card reader. Card reader checks $(T_re - T_se) \leq \Delta T_e$ and computes $Ae_1^* = hfu_1(ye_2, T_se)$, $Ae_2^* = hfu_1(Le_0, Ae_1^*)$, and $Ae_3^* = Ae_2.PWDe_x{}^*.RP1$. On successful matching of Ae_3^* and Ae_3, $ssKe = hfu_1(Ae_2^*, Le_1, Ae_1^*, Le_2)$ is calculated.

3.1.5 Password change phase

On the successful entry of User ID and password, new password is provided by the user for the new registration.

3.2 Working of *hrngRO*

The end user entered variable sized alpha numeric string with special characters is fed to the SHA-256 algorithm. The fixed sized hash value that comes out from SHA-256 algorithm is fed as a challenge to the Ring Oscillator PUF module. The response from the RO-PUF module is a random number required for registration and authentication phases of *eUASBP*. In the registration phase password entered by the user is given as a input to the SHA-256 algorithm to generate the random number. During the authentication phase, the same password entered by the user is utilized to regenerate the random number and random transaction password entered by the user is utilized to generate another random number for the particular transaction. Preliminary user credential checking is done by using bloom filters in this paper. User entered userid and password are double hashed by using SHA-256 algorithm and RO-PUF. Hamming distances between those two values are calculated and stored into the 128-bit bloom filter by calculating the corresponding bit index during the registration phase. The same process is done during the authentication phase to check the presence of hamming distance in the bloom filter to ensure preliminary checking of userid and password. The presence of hamming distance ensures correct entry of userid and password. The absence of hamming distance in the bloom filter ensures wrong entry of userid and password to avoid unnecessary delay in the authentication process Sangeetha and Ramasubramanian (2015) Xue et al. (2020). SHA-256 algorithm is a one-way hash function and it generates a fixed size 256-bit hash value Yoshida and Biryukov (2005). SHA 256 algorithms can be used as a pseudo-random number generator(PRNG) Akram et al. (2012).

3.3 *hrngRO* authentication protocol phases

hrngRO involves five phases as followed, Initialization phase, User's registration phase, User's login phase, User's authentication phase, and Password change phase.

3.3.1 Initialization phase

Chooses cyclic groups G_{r1x}, G_{r2x}, bilinear map $\hat{e} : G_{r1x} x G_{r2x}$, random generator $RP_x \in G_{r1x}$, cryptographic hash function $hf_{1x} : \{0,1\}^* \rightarrow Z_{qx}^*$ and a secret key sK_x. Computes $P_{pubx} = sK_x.RP_x$, bilinear map $\hat{e}(RP_x, RP_x) = \propto$ and publishes all parameters as public except sK_x. It is depicted in the Fig. 3.

Initialization Phase

Remote Server (RS)

Chooses
- A multiplicative cyclic group G_{r2x} and An additive cyclic group G_{r1x}
- An admissible bilinear map $\hat{e} : G_{r1x} x\, G_{r2x}$
- A random generator $RPx \in G_{r1x}$
- A cryptographic hash function $hf_{1x}: \{0,1\}^* \rightarrow Z_{qx}^*$
- A secret key sKx

Computes
- Public key as $P_{pubx} = sKx.RPx$
- $\hat{e}(RPx, RPx) = \alpha$

Publishes public parameters
- $< G_{r1x}, G_{r2x}, \hat{e}, \alpha, RPx, qx, P_{pubx}, hf_{1x}(.) >$

Fig. 3 *hrngRO* - Initialization phase

3.3.2 User's registration phase:

User enters ID_{xx} and password PW_{xx}. Registration for preliminary user credential checking is done by the following steps. ID_{xx} and PW_{xx} are double hashed through SHA-256 and RO-PUF modules. Hamming distance between these double hashed values are calculated and the corresponding bit index is stored into the 128-bit BF. PW_{xx} is double hashed through SHA-256 module and RO-PUF module to generate a random cardinal y_x. User sends userid ID_{xx} and a hashed password PWD_{xx}(Uses random cardinal y_x). Remote Server (RS) computes R_{xx}, REG_{IDxx}, and stores the following in the smart card. $< R_{xx}, REG_{IDxx}, hf_{1x}(.), RP_x, P_{pubx} >$. It is depicted in the Fig. 4.

3.3.3 User's login phase

User enters ID_{xx} and PW_{xx}, card reader does preliminary user credential checking by calculating bit - index as mentioned in the registration phase. If bit-index matches with the BF content, proceeds with the next step. Else prompts the user to re - enter ID_{xx}, PW_{xx}. Computes PWD^*_{xx}, L_{0x}, and REG^*_{IDxx}. Matches REG^*_{IDxx} and REG_{IDxx}. Random transaction password entered by the user is utilized to generate a random cardinal $y1_x$ by the use of SHA-256 and RO-PUF modules. On successful match, card reader computes the following L_{1x}, L_{2x}, L_{3x}, L_{4x} and sends $< ID_{xx}, R_{xx}, L_{3x}, L_{4x}, T_{xx} >$ to the RS. It is depicted in the Fig. 5.

3.3.4 User's authentication phase

RS checks the format of ID_{xx} and the value of ΔT. Computes A_{0x}, PWD_{xx}, $L_{1x}{}'$, $L_{2x}{}'$. On successsful matching of $\widehat{e}(L_{3x}, RPx)$ and $\propto^{(sKx.(A_{0x}+L'_{2x}))}$, RS computes A_{1x}, A_{2x}, A_{3x}, y_{3x}, ssK_x by choosing a random cardinal y_{2x}. RS sends $< A_{3x}, y_{3x}, T_{sx} >$ to the card reader. Card reader checks ΔT and computes the following A^*_{1x}, A^*_{2x}, and A^*_{3x}. On successful matching of A^*_{3x} and A_{3x}, ssK_x is calculated. It is depicted in the Fig. 6.

3.3.5 Password change phase

On the successful entry of User ID and password, new password is provided by the user for the new registration.

3.4 Random number generation

Physical unclonable functions (PUF) generate a unique response based on physical entropy and so it is called as a digital fingerprint. PUFs can be broadly classified as electronic PUFs and non - electronic PUFs. This paper uses a delay-based electronic PUF called ring oscillator (RO)PUF. RO-PUFs are formed by connecting the odd

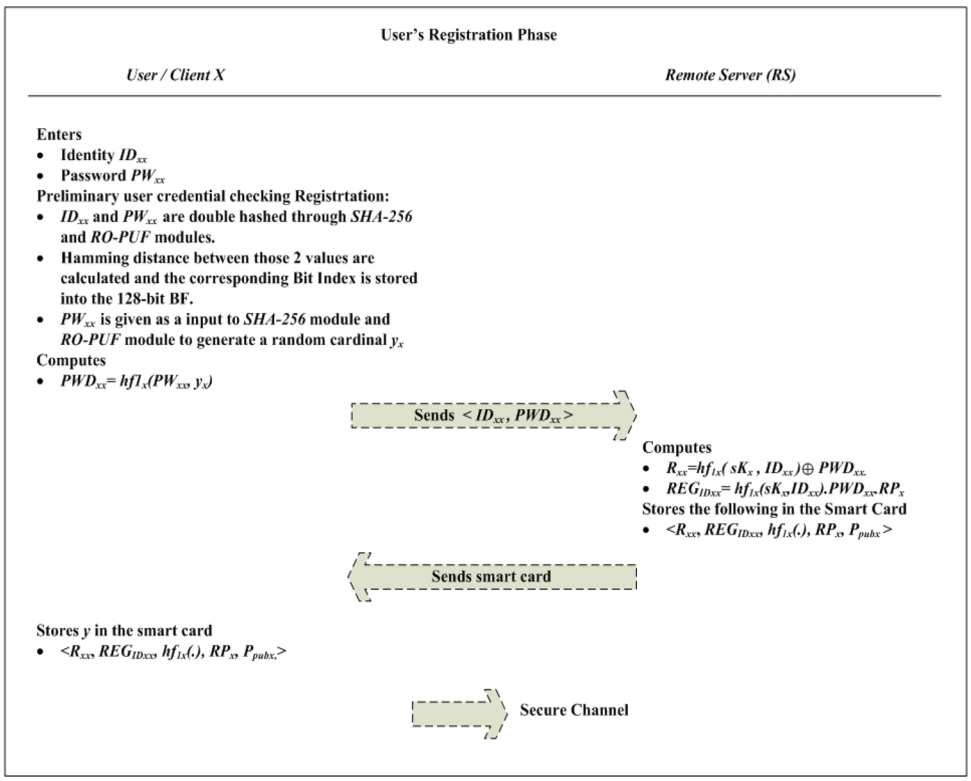

Fig. 4 *hrngRO* - User's registration phase

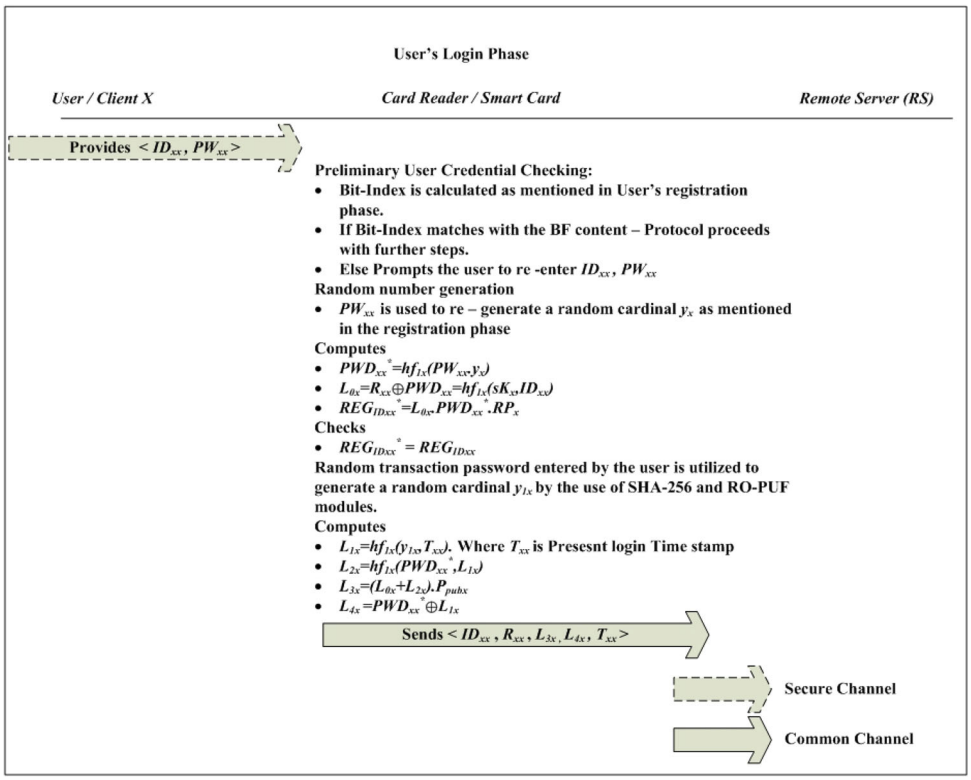

Fig. 5 *hrngRO* - User's login phase

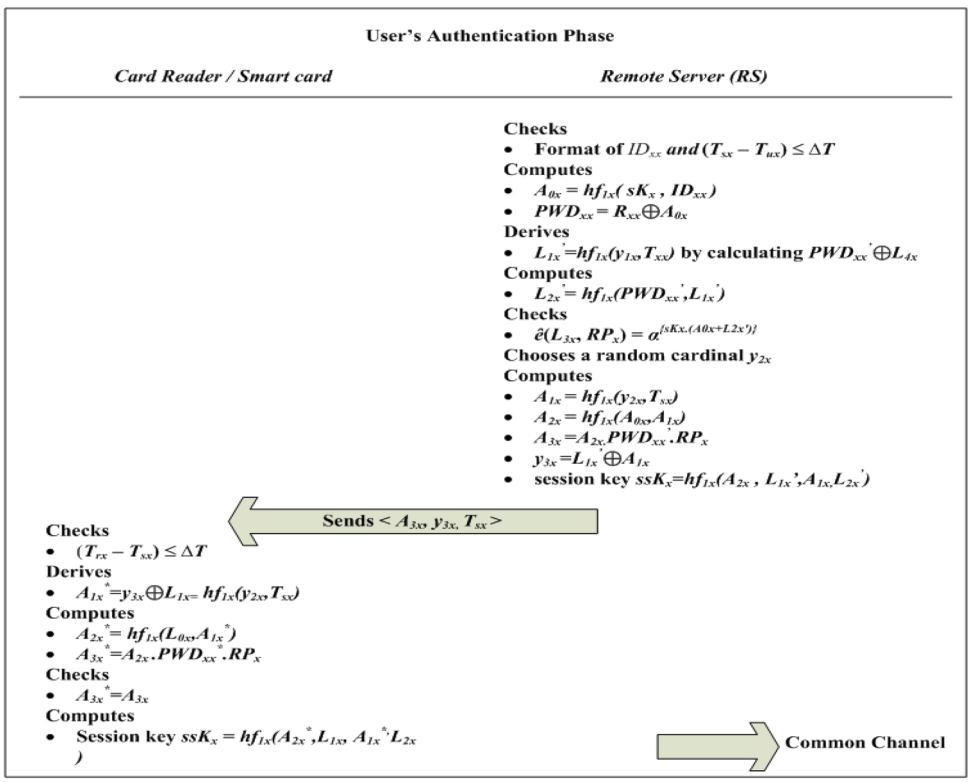

Fig. 6 *hrngRO* - User's authentication phase

number of inverters in a feedback loop that oscillates the input signal. The number of oscillations per time unit differs for every implementation of RO-PUF McGrath et al. (2019) Kokila et al. (2019) Kokila and Ramasubramanian (2019). This RO-PUF based random number generation is called as TRNG (True Random Number Generation). The

combination of SHA-256 algorithm (PRNG) and RO-PUF (TRNG) can be called as hybrid random number generator(HRNG) Goettfert et al. (2013). User entered variable size values are given as an input to SHA-256 module that generates 256-bit value. Generated 256 bit values are divided into two parts and given as S0 and S1 values for the RO-PUF module. In turn, RO-PUF module generates the 128-bit random value or response. This hybrid combination works like a double one-way hash function and improves security. So *hrngRO* can be called as HRNG. The implementation of SHA-256 algorithm is suggested to be kept in the card reader machine. And the implementation of RO-PUF is to be kept in the smart card as per *hrngRO*. Bloom filter is a data structure used to ensure the membership of an element in a set. It stores 1 or 0 in the calculated bit indices using any of the hash functions.

3.5 SHA-256 module

The working of SHA-2 or SHA-256 algorithm is depicted in Fig. 7. The input of this algorithm is of variable size. In the preprocessing or padder unit, variable-sized inputs are padded to make it a fixed size for processing. Message blocks are segmented and padded as required. Message blocks are scheduled in the scheduler unit. In the digest or compression unit, repeated compression for all the message blocks are being done and updations are done from temporary variables into the final hashed output variable. Control unit controls all the other units Cao and O'Neill (2012) Thakur and Malviya (2018).

3.6 RO-PUF module

The RO-PUF unit used in this paper contains eight RO elements. Every RO element contains an AND gate followed by five NOT gates and the last NOT gate connected to the AND gate as feedback. Triggering enable signal is shared by all the RO elements. The first four RO elements are connected by a multiplexer (MUX1) that uses s0, s1 as selection bits to select any one of the RO element among those four. In a similar manner, the next four RO elements are also connected (MUX2). The oscillations of selected RO elements are counted through counters 1 and 2. The counts c1 and c2 from counters are compared in the comparator to produce a single bit output. If c1 is greater than c2 then output is 1 else the output is 0. The working of RO-PUF is depicted in the Fig. 8.

3.7 BloomFilter module

In this work, 128-bit bloom filter is proposed for the preliminary credential checking of *eUASBP*. 128-bit bloom filter needs 7-bit bit indices. For the presence of a particular value 1 is stored and for the absence of a particular value 0 is stored. Bloom filters support only false positives and never support false negatives Sangeetha and Ramasubramanian (2015).

3.8 Security analysis module

Formal security analysis module is implemented using AVISPA tool. Authentication between Remote Server and User X of *eUASBP* is written in HLPSL with the following roles RemoteSer, UserX, session, environment. It is depicted in the following Figs. 9, 10, 11.

4 Implementation and results

hrngRO implementation has three major parts. They are the SHA-256 module, RO-PUF module, and BloomFilter module. This paper ensures the security of *eUASBP* using AVISPA tool.

4.1 SHA-256 module

Existing SHA-256 module implementation is taken for this work from [37]. The alphanumeric string with special characters of variable size is fed to SHA-256 module that produces 256-bit fixed size hashed value. SHA-256 algorithm is secure and unbroken unlike SHA-1 and MD5 hashing algorithms. SHA-256 is widely accepted and trusted by public sector units and industries. Supported 2^{256} possible hash values make collision impossible and generates different hash values even for a minor change in the original information. The implementation of SHA-256 in

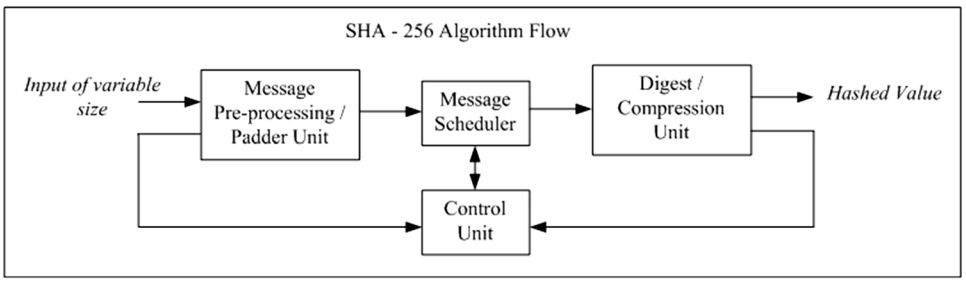

Fig. 7 SHA-2 or SHA-256

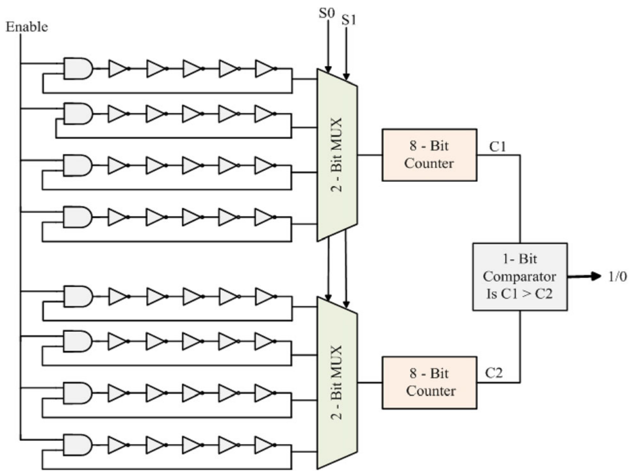

Fig. 8 RO unit

```
role RemoteSer( RS1,US1 : agent, K:public_key, Ari,Hash1 :
hash_func,
SND,RCV : channel(dy))
played_by RS1 def=
local State : nat,IDx,PWDx,RP:
text,Tx,Ts,Rx,L0,L1,L2,L3,A3,A1,A2,M2,Y3,L4,SK : message
init State := 0
transition
1. State = 0 /\ RCV(start) =|> State':= 2 /\
RCV({IDx.Rx.L3.L4.Tx}_inv(K)) /\ A2':= new() /\ RP' := new() /\
PWDx' := new() /\ A3':= Ari(A2.PWDx.RP) /\ Y3':= xor(L1,A1) /\
Ts':= new() /\ L2':= new() /\ SK' := Hash1(A2,L1,A1,L2') /\
SND(A3',Y3',Ts) /\ secret(SK',sk,{RS1,US1})
end role
```

Fig. 9 AVISPA implementation - remoteserver

FPGA is increasing the performance and the power utilization is reduced as stated in ref Kammoun et al. (2020).

4.2 RO-PUF module

RO-PUF module is implemented using Verilog HDL in Xilinx ISE 9.2i. 256 bit hashed value from SHA-256 module is fed as a challenge to the RO-PUF module. The 10 sample variable-size inputs used to take results are given below in Table 2. The corresponding 256 bit hashed values and the 128 - bit random numbers generated by the RO-PUF module are also given below in Table 2.RO-PUF is easy to be implemented for the industry standards but with more hardware components to implement. Random number generation of *;"1asqw!* using RO-PUF in Xilinx ISE 9.2i is given in the below Fig. 12. RTL schematic diagrams of 1-bit RO-PUF and 128-bit RO-PUF is given below in Fig. 13.

```
role UserX( RS1,US1 : agent,K:public_key, Ari,Hash1 : hash_func,
SND,RCV : channel(dy))
played_by US1 def=
local State : nat,IDx,PWDx,RP:
text,Tx,Ts,Rx,L0,L1,L2,L3,A3,A1,A2,M2,Y3,L4,SK: message %
same as RemoteSer
init State := 1
transition
1. State = 1 /\ RCV(start) =|> State':= 3 /\ L0':=new() /\ L1':=new() /\
L2':=new() /\ PWDx' := new() /\ L3':=Ari({L0'.L2'}_inv(K)) /\
L4':=xor(PWDx',L1') /\ SND({IDx.Rx.L3.L4.Tx}_inv(K)) /\
RCV(A3,Y3,Ts) /\ SK' := Hash1(A2,L1,A1,L2) % /\
secret(SK,sk,{RS1,US1})
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role session(RS1,US1 : agent,K:public_key,Ari, Hash1 :
hash_func)
def=
local SA, SB, RA, RB : channel (dy)
composition
alice(RS1,US1,K,Ari,Hash1,SA,RA) /\ bob
(RS1,US1,K,Ari,Hash1,SB,RB)
end role
```

Fig. 10 AVISPA implementation - user X and session

```
role environment()
def=
% declares session scenario
const
us1_rs1_a3,sk: protocol_id,
k:public_key,
rs1,us1 : agent,
ari,hash1 : hash_func,
idx : text
% specify what the attacker (aka. intruder) knows initially
intruder_knowledge = {rs1,us1}
% specify protocol scenario: how many sessions run in parallel
% and in which instantiations
composition
session(rs1,us1,k,ari,hash1)
% normal session
/\ session(rs1,i,k,ari,hash1)
% intruder i impersonates b
/\ session(i,us1,k,ari,hash1)
% intruder i impersonates a
end role
goal
secrecy_of sk
authentication_on us1_rs1_a3
end goal
environment()
```

Fig. 11 AVISPA implementation - environment

4.3 BloomFilter module

If we assume *meena_78_2_1* as a username and *;"1asqw!* as a password entered by the user during the registration phase. The corresponding random numbers generated will be 3B7BAE8315825B61 1987FF5238037799,

Table 2 *hrngRO* I/O flow

Password/Alphanumeric string with special characters	SHA-256 output		128-bit Random number
	128-bit S0	128-bit S1	
meena_78_2_1	7688B6EF52555962 D008FFF894223582	C484517CEA7DA49E E67800ADC7FC8866	3B7BAE8315825B61 1987FF5238037799
%opera_ty67	759B87B87BA0C0C7 01D14EB2E6E31560	929E935915378188 56D079634EE1BB68	6D616CA6EAC87E77 A92F869CB11E4497
?90namas111111111111111	6B86B273FF34FCE1 9D6B804EFF5A3F57	47ADA4EAA22F1D49 C01E52DDB7875B4B	B8525B155DD0E2B6 3FE1AD224878A4B4
User"lamp \/$45912603-	35135AAA6CC23891 sB40CB3F378C53A17	A1127210CE60E125 CCF03EFCFDAEC458	5EED8DEF319F1EDA 330FC10302513BA7
kk&pudur#123$san	8722616204217EDD B39E7DF969E0698A	ED8E599BA62ED2DE 1CE49B03ADE0FEDE	1271A66459D12D21 E31B64FC521F0121
*tms ∧ nithya**	21EF779311A43F0E 067D0F4F600BB545	1A8A7E093662086A 1FE6A75D27D7892A	E57581F6C99DF795 E01958A2D82876D5
;"1asqw!	0604CD3138FEED20 2EF293E062DA2F47	20F77A05D25EE036 A7A01C9CFCDD1F0A	DF0885FA2DA11FC9 585FE3630322E0F5
abcd%45(12)&qoqqqqqooo	BDD2D3AF3A5A1213 497D4F1F7BFCDA89	8274FE9CB5401BBC 0190885664708FC2	7D8B01634ABFE443 FE6F77A99B8F703D
——____32lopqwww	83EAF4DC5E19BCBE B23801E2C3E08C4A	89CC82D0A42A9037 67F9C938D1DEAC4F	76337D2F5BD56FC8 980636C72E2153B0
dash ∗ @ ∧ $()!	37834F2F25762F23 E1F74A531CBE445D	B73D6765EBE60878 A7DFBECD7D4AF6E1	48C2989A1419F787 5820413282B5091E

DF0885FA2DA11FC9 585FE3630322E0F5. And the hamming distance between them is 57. The corresponding bit index is 0111001. Value 1 is stored in the bit index 0111001. During the authentication phase, on entering the username and password, the random numbers are generated and the hamming distance is calculated to find the bit index. If value 1 is present in that bit index, user-entered credentials are correct else the user is prompted to re-enter the credentials. Hamming distance calculation between userid and password is depicted in the following Table 3. Bloom filters are fixed in size irrespective of the number of inputs. Bloom filters support false positives and never support false negatives

5 Security analysis

Formal and informal security analysis of *hrngRO* is being done to ensure its effectiveness and protection against attacks. Formal security analysis is being performed by the use of AVISPA tool. Informal security analysis discusses, most of the possible password-based smart card attacks and gives a tabulated form of security analysis that compares with the other state-of-the-art protocols. Following security analysis proves that *hrngRO* protocol provides better protection than other related protocols.

5.1 Formal security analysis of *hrngRO*

Formal security analysis of *hrngRO* is done by using AVISPA tool. The formal model of Authentication between Remote Server and End User of *hrngRO* is written in HLPSL. And those models are analyzed and verified using AVISPA. Authentication between Remote Server and End User of *hrngRO* is written in HLPSL with the following roles (Remote Server)RemoteSer, (End User)-UserX, session, and environment. The corresponding HLPSL codes are presented in section 3.8. And tested under the backends OFMC, and CL-AtSe. The corresponding results shown in the Figs. 14, and 15, ensure the safety of the protocol under analysis and attacks. In the OFMC model output and CL-AtSe model output, SUMMARY generated is SAFE that indicates the security strength of the proposed work

5.2 Informal security analysis of *hrngRO*

Weak random number generators open the doors for attacks and adversaries. Hybrid random number generator module and bloom filter module make *hrngRO* more strong and protect the system against attacks and adversaries. *hrngRO* specifically protects from the following attacks and provides Pre Early Wrong Password Detection.

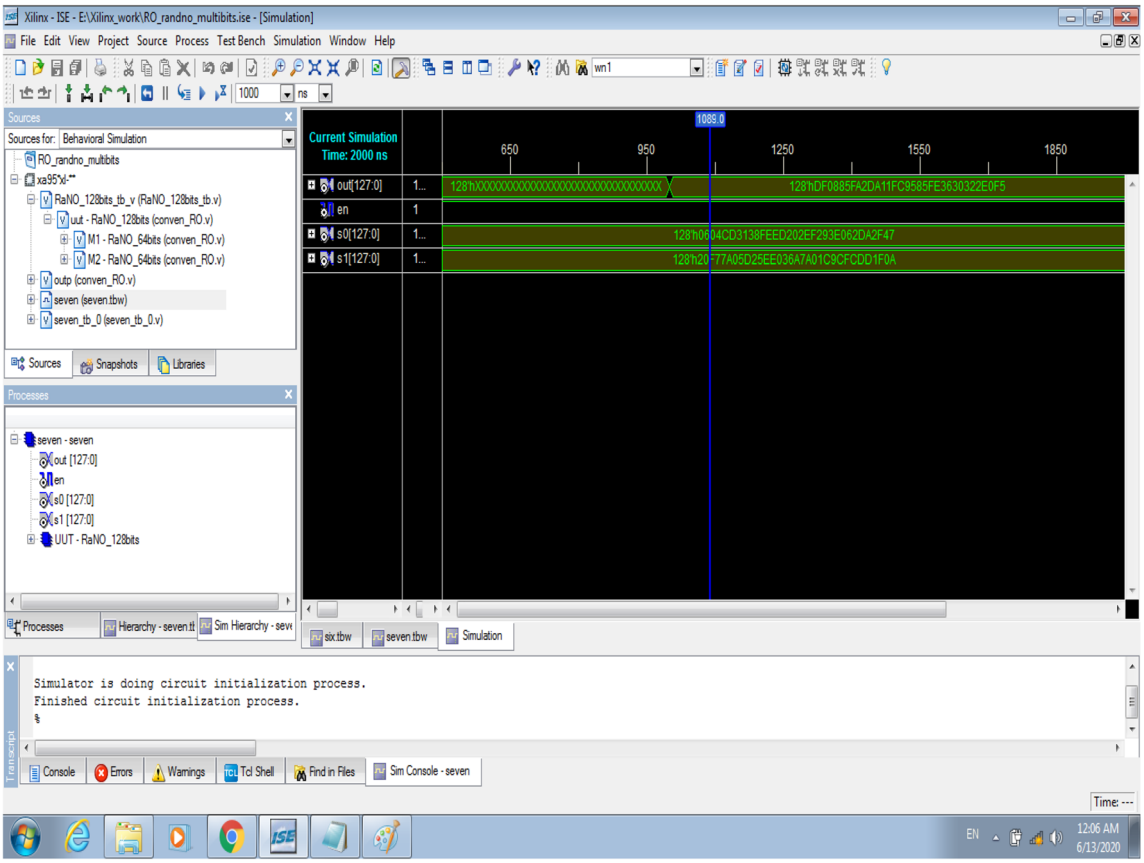

Fig. 12 RNG generation using RO-PUF

5.2.1 Pre early wrong password detection

Wrong entry of username and password is detected by the use of bloom filter, before proceeding with the steps of authentication protocol. That is the hamming distance between true random cardinals of username and password as per *hrngRO* is matched against bloom filter present in the smart card. If it matches, it proceeds with the authentication protocol steps else intimates the user to re-enter the username and password. Pre Early Wrong Password Detection avoids the execution of authentication protocol on wrong entry of username or password, thereby saves the remote server's execution time.

5.2.2 Crypt analytic attack

SHA-256 algorithm and RO-PUF act as one-way hash functions. Inversion property of those cryptographic one-way hash functions leads to double inversion property of *hrngRO*. SHA-256 module generates pseudo-random cardinals on inputting seed values. RO-PUF module generates true random cardinals on inputting pseudo-random cardinals from the SHA-256 algorithm. This double inversion property makes Crypt Analytic attack impossible with *hrngRO*.

5.2.3 Cloning attack

It becomes impossible with *hrngRO* because of the physical entropy property of RO-PUF and inversion property of the SHA-256 algorithm. Cloning a smart card leads to different physical entropy properties and different random values generation.

5.2.4 Reverse engineering attack

Backtracking the result, design, or implementation to reproduce the smart card is called a reverse engineering attack. It becomes impossible with *hrngRO* because of the physical entropy property of RO-PUF.

5.2.5 Session key discloser attack

Session key's security is ensured by the secret parameters $< sK_x, PWD_{xx}, y_{1x}, y_{2x} >$ and cryptographic hash function. Obtaining the secret parameters becomes impossible as

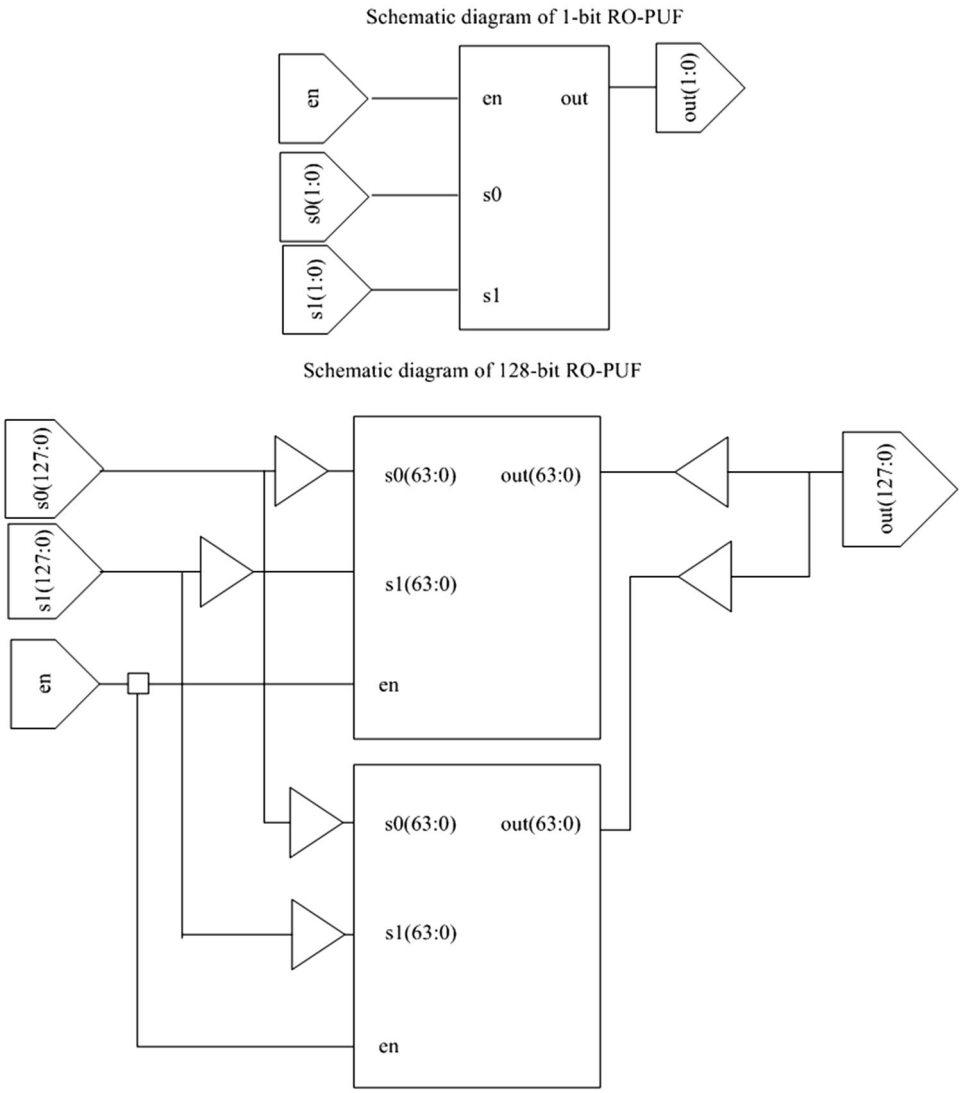

Fig. 13 RTL schematic diagram of 1-bit RO-PUF and 128-bit RO-PUF

Table 3 *hrngRO* Hamming Distance Calculation

User Id	User Id Random no.	Password	Password Random no.	Hamming Distance	Bit Index
meena_78_2_1	3B7BAE8315825B61 1987FF5238037799	*%opera_ty67*	6D616CA6EAC87E77 A92F869CB11E4497	60	0111100
?90namas111111111111111	B8525B155DD0E2B6 3FE1AD224878A4B4	*User"lamp \/$ 45912603-*	5EED8DEF319F1EDA 330FC10302513BA7	70	1000110
kk&pudur#123$san	1271A66459D12D21 E31B64FC521F0121	*tms ∧nithya**	E57581F6C99DF795 E01958A2D82876D5	60	0111100
;"1asqw!	DF0885FA2DA11FC9 585FE3630322E0F5	*abcd%45(12)&qo qqqqqooo*	7D8B01634ABFE443 FE6F77A99B8F703D	59	0111011
——____32lopqwww	76337D2F5BD56FC8 980636C72E2153B0	*dash*@ ∧$() !~*	48C2989A1419F787 5820413282B5091E	70	1000110

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/Auth_RS_US.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.01s
  searchTime: 0.20s
  visitedNodes: 16 nodes
  depth: 4 plies
```

Fig. 14 OFMC model

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/Auth_RS_US.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 0 states
  Reachable  : 0 states
  Translation: 0.05 seconds
  Computation: 0.00 seconds
```

Fig. 15 CL-AtSe model

mentioned in the other attacks. So, session key discloser attack becomes impossible with *hrngRO*.

5.2.6 Smartcard stolen attack

Through stolen/missed card, private values stored in the smart card can be retrieved. i_{th} instance of the login message can be obtained by the adversary but valid login message cannot be made without knowing y_{3x}, y_{2x}, sK_x and PW_{xx}. It is impossible to derive the mentioned values, so smartcard stolen attack is not possible with *hrngRO*.

5.2.7 Server masquerading attack

To imitate a remote server, the adversary needs to make a forged reply message for the copy of the user's login request message. Hard to make $< A_{3x},\ y_{3x} >$ reply message by not knowing sK_x, PW_{xx}, and y_{2x}. As mentioned in the other attacks, stated values cannot be derived. So, server masquerading attack is not possible with *hrngRO*.

5.2.8 Theft attack

To perform successful theft attack, adversary is in need of valid registration parameters. It is not possible to derive R_{xx} or REG_{IDxx}. So theft attack is impossible with *hrngRO*.

5.2.9 User impersonation attack

Adversary need to make a forged login message, to act as an actual user/client with the RS. Valid login message $< ID_{xx},\ R_{xx},\ L_{3x},\ L_{4x},\ T_{xx} >$ cannot be made by the adversary. Without knowing the values of PW_{xx}, y_{1x}, sK_x adversary cannot make a valid login message. PW_{xx} cannot be obtained by listening to the communication channel or by retrieving values from the smart card memory. So, user impersonation attack becomes impossible with *hrngRO*.

5.2.10 Privileged insider attack

PW_x cannot be derived from PWD_x. Since it is communicated in the hashed form. Inversion property of cryptographic one-way hash function makes it impossible. So *hrngRO* is protected against privileged insider attack.

5.2.11 Offline - password guessing attack

This attack can be performed by fetching stored values from smart card or by fetching values from communication channel. Case1: Values stored in the smart card are $< R_{xx}, REG_{IDxx}, hf_{1x}(.), y_x >$. If adversary tries to guess password by manipulating R_{xx}, he will be in need of $< sK_x$ of RS, that is 128 bits long. If adversary tries to guess password by manipulating REG_{IDxx}, Discrete Logarithm Problem has to be solved that is not solvable in polynomial time. Case2: Adversary can retrieve login message $< ID_{xx}, R_{xx}, L_{3x}, L_{4x}, T_{xx} >$ and reply message $< A_{3x},\ y_{3x},\ T_{sx} >$ from the communication channel. In order to guess password adversary need to know sK_x and y_{3x} that are intractable in polynomial time.

6 Conclusion

Proposed *hrngRO* provides Ring Oscillator PUF based random number generator using SHA256 hash function. This hybrid combination of RNG makes *hrngRO* more secure against many other attacks like Crypt analytic attack, Cloning attack, Reverse engineering attack and makes the system even faster. Pre Early Wrong Password

Table 4 Comparison of attacks possibility of proposed *hrngRO* with related schemes

Schemes	eUASBP Rajaram et al. (2019)	Giri and Srivastava (2006)	Manik et al. Das et al. (2006)	Bayat et al. (2010)	Awasthi′s Awasthi (2012)	Fang and huang (2006)	Jia et al. (2006)	*hrngRO*
At1	✔	X	X	✔	X	X	X	✔
At2	✔	X	X	✔	X	X	X	✔
At3	✔	X	X	✔	X	X	X	✔
At4	✔	X	X	✔	X	X	X	✔
EPD	✔	X	X	X	✔	X	X	✔
MA	✔	X	X	✔	X	X	X	✔
SK	✔	X	X	X	X	X	X	✔
At5	X	X	X	X	X	X	X	✔
Cl	X	X	X	X	X	X	X	✔
RE	X	X	X	X	X	X	X	✔
pEPD	X	X	X	X	X	X	X	✔

At1: Resist insider attack, At2: Resist user impersonation attack, At3: Resist smartcard stolen attack, At4: Resist off-line password guessing attack, At5: Resist Crypt Analytic attack, EPD: Early wrong password detection, MA: Mutual authentication, SK: Session key agreement, Cl: Resist cloning attack, RE: Reverse Engineering, pEPD: pre Early wrong password detection, ✔: Can be mounted or satisfied, X: Cannot be mounted or not satisfied

Detection is supported by *hrngRO* using bloom filters. *hrngRO* is secure enough than other related protocols. The proposed hybrid RNG module can be used in most of the authentication protocols present in the market. Future direction is to analyze the performance of this protocol based on computation and communication costs.

Funding No funding is provided for this work.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Human and animals rights This article does not contain any studies with human participants or animals performed by any of the authors

Informed consent Informed consent was obtained from all individual participants included in the study.

References

https://passwordsgenerator.net/sha256-hash-generator/

Akram RN, Markantonakis K, Mayes K (2012) Pseudorandom number generation in smart cards: an implementation, performance and randomness analysis. In: 2012 5th International Conference on New Technologies, Mobility and Security (NTMS), pages 1–7. IEEE

Awasthi AK (2012) An improved remote user authentication scheme with smart cards using bilinear pairings. Int J Appl Math Comput, 4(4)

Bayat M, Sabzinejad M, Movahed A (2010) A novel secure bilinear pairing based remote user authentication scheme with smart card. In: Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP 8th International Conference on, pages 578–582. IEEE,

Bond M, Choudary MO, Murdoch SJ, Skorobogatov S, Anderson R (2015) Be prepared: the emv preplay attack. IEEE Security Privacy 13(2):56–64

Cao X, O'Neill M (2012) Application-oriented sha-256 hardware design for low-cost rfid. In: 2012 IEEE International Symposium on Circuits and Systems, pages 1412–1415. IEEE,

Chen E, Ye Z, Wang C, Mingtao X (2019) Subway passenger flow prediction for special events using smart card data. IEEE Trans Intell Transp Syst 21(3):1109–1120

Chou J-S, Chen Y, Lin J-Y (2005) Improvement of manik et al.!'s remote user authentication scheme. IACR Cryptol. ePrint Arch., 2005:450

Das ML, Saxena A, Gulati VP, Phatak DB (2006) A novel remote user authentication scheme using bilinear pairings. Comput Security 25(3):184–189

El MN , Bertin E, Pujolle G (2018) An overview of the emv protocol and its security vulnerabilities. In *2018 Fourth International Conference on Mobile and Secure Services (MobiSecServ)*, pages 1–5. IEEE

Fang G, huang G (2006) Improvement of recently proposed remote user authentication schemes. IACR Cryptology ePrint Archive 2006:200

Furnell S (2020) Passwords a lesson in cyber security failure? ITNOW 62(1):26–27

Giri D, Srivastava PD (2006) An improved remote user authentication scheme with smart cards using bilinear pairings. IACR Cryptology ePrint Archive 2006:274

Goettfert R, Rueping S, Gammel B (2013) Hybrid random number generator, November 26. US Patent 8,595,277

Goriparthi T, Das ML, Saxena A (2009) An improved bilinear pairing based remote user authentication scheme. Comput Standards Interf 31(1):181–185

Hakeem SAA, Kim H (2021) Multi-zone authentication and privacy-preserving protocol (mapp) based on the bilinear pairing cryptography for 5g-v2x. Sensors 21(2):665

He Y, Alem EE, Wang W (2020) Hybritus: a password strength checker by ensemble learning from the query feedbacks of websites. Front Comp Sci 14(3):1–14

Jia Z, Zhang Y, Shao H, Lin Y, Wang J (2006) A remote user authentication scheme using bilinear pairings and ecc. In: Intelligent Systems Design and Applications, 2006. ISDA'06. Sixth International Conference on, volume 2, pages 1091–1094. IEEE

Kammoun M, Elleuchi M, Abid , BenSaleh MS (2020) Fpga-based implementation of the sha-256 hash algorithm. In: 2020 IEEE International Conference on Design & Test of Integrated Micro & Nano-Systems (DTS), pages 1–6. IEEE

Kokila J, Das AM, Begum BS, Ramasubramanian N (2019) Hardware signature generation using a hybrid puf and fsm model for an soc architecture. Periodica Polytechnica Elect Eng Comput Sci 63(4):244–253

Kokila J, Ramasubramanian N (2019) Enhanced authentication using hybrid puf with fsm for protecting ips of soc fpgas. J Electron Test 35(4):543–558

Kou J, He M, Xiong L, Lv Z (2020) Efficient hierarchical authentication protocol for multiserver architecture. Security and Communication Networks, 2020,

Lara-Nino CA, Diaz-Perez A, Morales-Sandoval M (2020) Lightweight elliptic curve cryptography accelerator for internet of things applications. *Ad Hoc Networks*, page 102159

Lee T-F, Chen W-Y (2021) Lightweight fog computing-based authentication protocols using physically unclonable functions for internet of medical things. J Inf Security Appl 59:102817

Li W, Yan X, Li Xi, Yang J (2020) Estimate passengers' walking and waiting time in metro station using smart card data (scd). IEEE Access 8:11074–11083

Li G, Zeng Y, Guang H, Yu G (2020) A priority-aware anonymous handover authentication protocol for wireless communications. wireless personal commun

McGrath T, Bagci IE, Wang ZM, Roedig Utz, Young RJ (2019) A puf taxonomy. Appl Phys Rev 6(1):011303

Rajaram S, Maitra T, Vollala S, Ramasubramanian N, Amin R (2019) euasbp: enhanced user authentication scheme based on bilinear pairing. J Ambient Intel Humanized Comput, pages 1–14,

Raponi S, Di PPietro R (2020) A longitudinal study on web-sites password management (in) security: evidence and remedies. IEEE Access 8:52075–52090

Sangeetha R, Ramasubramanian N (2015) A survey of hardware signature implementations in multi-core systems. In: 2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN), pages 1–5. IEEE

Shamshad S, Mahmood K, Kumari S (2020) Comments on a multi-factor user authentication and key agreement protocol based on bilinear pairing for the internet of things. Wireless Personal Communications, pages 1–4,

Sun H-M, Chen Y-H, Lin Y-H (2011) opass: a user authentication protocol resistant to password stealing and password reuse attacks. IEEE Trans Inf Forensics Secur 7(2):651–663

Thakur D, Malviya U (2018) Low power and simple implementation of secure hashing algorithm (sha-2) using vhdl implemented on fpga of sha-224/256 core. Int J Eng Manag Res (IJEMR) 8(1):1–4

Tseng Y-M, Tsu-Yang W, Jui-Di W (2008) A pairing-based user authentication scheme for wireless clients with smart cards. Informatica 19(2):285–302

Xue W, Vatsalan D, Wen Hu, Seneviratne A (2020) Sequence data matching and beyond: New privacy-preserving primitives based on bloom filters. IEEE Trans Inf Forensics Secur 15:2973–2987

Yoshida H, Biryukov A (2005) Analysis of a sha-256 variant. In: International Workshop on Selected Areas in Cryptography, pages 245–260. Springer

Zhang Y, Cheng T (2019) A deep learning approach to infer employment status of passengers by using smart card data. IEEE Trans Intell Transp Syst 21(2):617–629

Zhu F, Li P, He X, Wang R (2019) A lightweight rfid mutual authentication protocol with puf. Sensors 19(13):2957