Memristors for Secret Sharing Based Lightweight Authentication

Md Tanvir Arafin, *Student Member, IEEE,* and Gang Qu, *Senior Member, IEEE*

***Abstract*—User authentication is one of the most fundamental security problems that design effective ways of identifying single or multiple entities using shared information, signatures or intrinsic properties of the users. Password-based authentication is standard in the computer systems; however, passwords usually have low entropy content, and therefore, vulnerable to dictionary attacks. Furthermore, password storage and simultaneous multi-party authentication also pose security and privacy concerns. Secret sharing based techniques during password enrollment are found to be helpful in securing key-storage in the authentication server and in assisting multi-party authentication without exposing individual identity. However, secret sharing techniques such as Shamir's secret sharing is computationally expensive, and therefore its implementation in power-constrained systems are elusive. To address these problems, we have demonstrated how secure and lightweight user authentication techniques can be designed using several well-known properties of memristive devices. For developing our secret sharing based computationally lightweight user authentication protocols, first, we define essential utility functions such as Read State, SET Pulse Count, Preconditioning *etc.* for controlling conductive filament formation in memristive devices. Next, we demonstrate the implementation of hardware dependent simple authentication protocols that can ensure secure key-storage using secret sharing protocol derived from Shamir and Naor's Visual Cryptographic constructs. Then, we lay out the required hardware design, and discuss the potential attacks to these protocols and the corresponding countermeasures. We conclude that – under realistic attacking assumptions, the proposed protocols are secure. Finally, using PTM's 65nm MOSFET models and Stanford's variation-aware memristor models, we perform HSPICE simulation of the secret-reconstruction and authentication units to demonstrate the reliability of the hardware designs against SET-RESET unbalance, noise, temperature fluctuations and aging.**

***Index Terms*—Memristor, User Authentication, Device Authentication, Secret Sharing, Hardware-Oriented Security, and Trust.**

I. Introduction

High-density memristive crossbar arrays are promising candidates for next-generation memory solutions with in-memory computing capabilities [1]. Simple device geometry, high density, and built-in non-volatile features have made memristors a replacement for flash memory and DRAM. However, commercial products based on memristors are still quite elusive. Therefore, for developing competitive products further research on the fabrication process, yield, novel computer architecture, and circuit design are required. Furthermore, applications outside memory operation such as security and cryptographic applications need to be explored to harness the unique characteristics of memristive devices. For example, time-dependent switching properties of metal-oxide-based resistive memories have demonstrated applications in Physically Unclonable Function (PUF) designs [2], [3]. These works have attracted new ideas and memristor-based research in the area of hardware-dependent security and trust.

Authors are with the Department of Electrical and Computer Engineering, University of Maryland, MD, 20742.

Hardware security is an emerging field that studies the application of security primitives in hardware and their vulnerabilities. As new technologies emerge, the nature of security primitives and vulnerabilities change, hence, considering security features of novel technologies have its unique benefits. Moreover, emerging technologies can provide exclusive hardware features useful for security purpose. Instead of making hardware the weakest link in security via side channel and probing attacks, one can employ hardware's intrinsic properties to improve the cryptographic and non-cryptographic applications.

In the advent of memristors for memory design and in-memory computation, works exploring memristor-based security primitives such as physically unclonable functions [4], [2], random number generators [5] and chaotic circuits for secure communication [6], [7] *etc.* are also gaining research attention. This work is a continuation of such efforts in the area of security revolving user and device authentication techniques.

Common authentication methods using low-entropy user-passwords are weak and severely vulnerable to dictionary attacks. Although password-authenticated key exchange protocols can render secure authentication against active and passive attackers, they are susceptible to dictionary attacks at the authentication server. Secret sharing schemes can be useful in creating and distribution authentication secrets over multiple servers for resisting dictionary attacks at the server-side. However, general secret sharing mechanisms such as the Shamir's secret sharing algorithm and its derivatives are hardware independent and require significant power and computation capabilities. Therefore, with the advent of distributed Internet-of-Things (IoT) systems, authentication would become one of the standard security challenges for low-power IoT nodes. In this work, we have explored how the intrinsic physical properties of memristors can be useful in tackling this problem.

In this work, we address the problem of secure and lightweight authentication of single and multiple entities in a computationally constrained system. We assume an entity (say Alice) tries to simultaneously authenticate k-out-of-n-other entities $(B_1, \dots B_k)$. This problem can be simplified as simultaneous verification of k-out-of-n cryptographic keys (*i.e.,* passwords). If these keys are generated from a secret that Alice knows or possesses, then, instead of checking n-keys separately, Alice can use the keys provided by $B_1, \dots B_k$ to regenerate her secret and authenticate all entities at once. Note that, as Alice uses a k-out-of-n secret sharing scheme, Alice can reconstruct the secret from any of the k-users shares. The problem of authenticating a single user (say Bob) then reduces to a 1-out-of-1 authentication problem. For avoiding collusion among the users, this procedure requires Alice to have some interference in key/password generation process during the registration of the users.

Since our work focuses on a computationally constrained system, we consider using emerging memristor devices for secret sharing based user authentication. To the best of our knowledge, this is one of the first attempts at using memristive

hardware for such applications.The key contributions of this work can be considered as follows:

1) We design two single user authentication protocols and one multiple user authentication protocols using memristive hardware. This hardware implements the key constructs of Visual Cryptography based secret sharing for user authentication.
2) We discuss the potential attacks on these protocols and provide corresponding countermeasures for understanding the merits of the proposed schemes regarding security.
3) We perform HSPICE simulation using PTM's 65nm MOSFET model and Stanford RRAM model to validate the reliability of our approach.

The main advantage of our approach is that it does not need the computational expensive cryptographic techniques. In that sense, our approach can be considered as a lightweight cryptographic method for secret sharing based authentication. Intrinsic energy efficiency of memristive systems makes this even more attractive. In the next section, we provide a review of the current literary works in the field of authentication, secret sharing, and memristor-based hardware security designs. In section III, we give basics on memristor device physics and the standard physical properties of memristors that are useful in creating fundamental security primitives. Section IV discusses the proposed single and multiple user authentication protocols using memristive hardware. The corresponding hardware designs are presented in section V. Finally, section VI and VII illustrate simulation of the authentication protocols in the proposed hardware and reports the key reliability issues and their countermeasure.

II. Literature Review

Identity and entity authentication depends on the sharing a secret between two parties– the verifier and the claimant. In this section, we discuss the current state of hardware-dependent secret sharing and user authentication mechanisms.

A. Single User Authentication

For authenticating a single user, the verifier authenticates the singular claimant based on a secret that can be derived from (1) something that is known by both parties (such as passwords), or (2) something possessed by the claimant (such as hardware keys), or (3) something inherent (such as signatures, biometric signals *etc.* of the claimant) [8]. According to Menezes *et al.* common properties of authentication protocols includes "reciprocity of identification, computational efficiency, communication efficiency, nature of security guarantee and the nature of security storage" [8].

Passwords are the most commonly used user authentication mechanism where the authenticator stores the (username, password (or its hashed value)) pair for different claimant and use this pair to identify a claimant. To strengthen this protocol several steps can be taken such as (i) different password rules can be introduced, (ii) password salting can be performed, or (iii) password mapping can be slowed down which will make it difficult for an attacker to test a large number of trial passes [8]. Frequent attacks on password schemes are replay attack and exhaustive and dictionary password search. Furthermore, leaking of an authenticator's database containing (username, password) can cause significant threat [9].

For all of these weaknesses of password schemes, challenge-response identification becomes a step toward strong authentication where the authentic parties share a sequence of secret one time passwords or challenge response pairs which are usually derived from some one-way functions or a challenge-response table. Furthermore, a zero-knowledge protocol that verifies an entity through its possession of the knowledge of the secret instead the exact secret is also a strong authentication scheme [8].

B. Secret Sharing and Multiple User Authentication

Secret sharing is a well-studied problem, which seeks to distribute pieces of information (or called the secret) to multiple parties in a way such that the information (the secret) can be revealed when all or a sufficiently large subset of the individuals contribute their shares. Shamir's secret sharing algorithm [10] defines the concept of threshold scheme for secret sharing. This (k, n) threshold scheme (where $k \leq n$) can be described as follows:

Assume that a secret S needs to be shared by n parties. The secret is divided into n pieces such that, the knowledge of k or more pieces would be sufficient to reconstruct S. However, if the knowledge of any $k-1$ pieces or less is available, it would be impossible to restore S.

A direct application of this problem is the multi-user authentication problem, where at least k authentic users must present their shares to gain access to a system. One example of this is the "*two-person rule*" originally designed to prevent the accidental or malicious launch of nuclear weapons by a single individual. One may argue that authenticating each user and counting the authenticated users can trivially solve the multi-user authentication problem. This approach does address the issue but has two significant drawbacks in scalability and user privacy. First, the expensive authentication protocol has to be applied at least k times, and some mechanism to check duplicate users must be implemented. Second, unlike the traditional secret share dependent approaches [10]–[13], this method will reveal the identity of each authenticated user, creating user privacy concerns.

Shamir [10] and Blakley [11] independently proposed solutions for the aforementioned problem of (k, n)-threshold scheme. More specifically, Blakley used techniques from finite geometry to provide a solution for safeguarding and sharing cryptographic keys. Shamir's solution is designed on the polynomial interpolation over a finite field. It has since become a widely acceptable solution for secure secret sharing and distribution of cryptographic keys. A detailed explanation of the scheme and the impact of subsequent works can be found in [12]. One of the drawbacks of these early solutions is their high computational cost. For example, Shamir's secret sharing algorithm requires calculations over the finite field during both secret share generation phase and secret reconstruction phase. This, in turn, requires complex digital circuitry for hardware implementation, which is known to be more efficient than the software implementation.

A simple hardware-oriented secret sharing model was first proposed by Naor and Shamir [13] as the Visual Cryptography(VC) scheme. It is an alternative lightweight system for secret sharing to Shamir's original scheme. In visual cryptography, the secret shared among multiple parties is simply an image. The secret image is broken into n pieces, and each piece is printed on a transparency. When k or more pieces of these transparencies are placed on a stack, the secret image is revealed and can be comprehended by human eyes. In this scheme, the secret share generation needs only straightforward calculations, and the revelation of the secret image does not involve any mathematical computation. In essence, this is an example of how inherent physical properties of hardware can be used in, designing lightweight security primitives to avoid complex mathematical computation and formal cryptography

protocols. Naor and Pinkas first discussed the application of visual cryptography for authentication and identification [14].

C. Hardware in Authentication: PUFs for Key Generation and Storage

The first problem for authentication is key-generation. For solving this, a signature of Bob is required to be stored by the Alice to verify Bob. This procedure poses several security challenges. First, the human-generated keys such as passwords are rarely random and are vulnerable to dictionary attacks, the machine-generated hardware keys might be difficult to remember, and storing a key in non-volatile memory can leak the key to an adversary.

Physically Unclonable Functions [15]–[17] can provide some solutions to key generation and storage problem. Silicon PUFs are on-chip circuitry that can extract fabrication variations to generate chip-dependent PUF data that can be used as secret keys or as seeds of random number generators. For using PUFs in authentication, one can hash a password with the PUF signature to increase the entropy of the password and generate a unique device dependent key. Moreover, PUFs does not store the key; rather the key exists when the device is powered on, and therefore, to extract a key an adversary needs to attack the system while it is running, which is significantly harder [18]. Therefore, new design technologies such as resistive memory-based random number generators and PUFs can be useful in solving primary key generation and storage challenges.

D. Memristors in Security

Memristors can offer some unique advantages over traditional CMOS designs in developing hardware security primitives. Analog memristive devices and circuits can provide unique properties such as non-volatility, bias-dependent write-time, fabrication variations in a filament, and non-linearity in the current-voltage relationship [3]. These features are being investigated in the current literature to offer better hardware security design primitives. For example, a memristor is nonvolatile, and after formation or uneven write operation they can end up in a device dependent resistive state. SRAM-based PUFs are built on the somewhat similar principle of device dependent metastable region of the SRAM unit. However, the SRAM PUF implementation requires needs at least 6-transistors for generating 1 bit of entropy, whereas a memristive application uses only a 1-transistor and one memristor. Moreover, there is ongoing research on integrating memristive array in the CMOS process flow [19], [20], which would lead to faster on-chip memristive storage and easier integration of memristor-based security designs.

Efforts on nonvolatile memory based secure hardware designs are currently focused on the different categories, such as developing novel physically unclonable functions and random number generators, creating memristor based chaos circuits for secure communication, exploring secure memory architecture designs for non-volatile systems. There has been some progress in using memristive hardware for device authentication. For example, different design schemes of memristor/memristive physically unclonable functions(PUFs) can be found in current literature.

A public PUF, called nano-PPUF, is reported in [4]. It is built using memristor based crossbar array and supporting challenge-response circuits. It utilizes features including sneak path currents, process variations, and computationally intensive SPICE models. First, an accurate simulation model for the memristor crossbar is constructed using physical measurements. The challenge for this PUF can be built as an applied voltage in a given polyomino (a selection of specific memristors) in the crossbar, and the response can be measured from the output voltage or current. The fundamental idea is that although the complete physical model for the crossbar is known to the public, accurate simulation for a large polyomino is hard and therefore, without the physical crossbar and the knowledge of correct polyomino to choose, it would be computationally prohibitive to calculate the response for a given challenge under timing constraint. With a few hundreds, of memristors, the following time-bounded authentication protocol is implemented.

Discrete memristor based PUFs are also being proposed by several authors [4], [21]. For example, the PUF design offered by Rose *et al.* [2] uses the write-time variability of memristors. Write-time for a memristor can vary due to physical randomness, and by using a mean write-time one can introduce an uncertainty of the state of a given memristor. Therefore, a cell's physical unclonability is derived from its uncertain resistive state for a given write-time. Che *et al.* introduced a memristive PUF design that uses the random resistance distribution in a crossbar array as the source of entropy [22].

III. Memristive Properties and Requirements for Designing Security Constructs

A. Device Model

Memristors are fundamentally thin-film based metal-insulator-metal (MIM) devices. According to Chua, instantaneous resistance of a charge-controlled memristor can be written as [23]:

$$M(q) = \frac{d\phi(q)}{dq} \equiv \frac{v(t)}{i(t)} \tag{1}$$

where, $q = \int_{-\infty}^{t} i(\tau)d\tau$ is the electric charge, $\phi(q)$ is the magnetic flux, $v(t)$ is the voltage across the device, and $i(t)$ is the electric current flowing in the device at time t [23]. Therefore, from equation 1, we can see that the resistance of an ideal memristor is dependent on the current that has previously passed through the device.

Later, Kang and Chua generalized the concept of memristors and memristive system in [24]. It was theoretically argued that the dynamic properties of a current controlled memristive system can be expressed with the help of an internal state variable g as:

$$\frac{dg}{dt} = f(g, i) \tag{2}$$

$$v(t) = M(g, i) \times i(t) \tag{3}$$

where, $M(g, i)$ is the generalized resistance of the memristive system and $f(g, i)$ is a function that captures the boundary behavior and various nonlinear dynamical effects.

In this work, we have demonstrated our experiments on the HfO_x based memristors also known as resistive random access memories (RRAMs). These devices have an individual design and marked differences from the TiO_2 based memristors developed by HP Labs. HfO_x based memristive devices have large ON/OFF ratio, multi-bit storage capacity and excellent switching endurance ($\geq 10^6$ cycles) [1], and therefore, they are promising candidates for memristive designs. Although our experiments are devised and performed on this device platform, our theoretical constructs for secret-sharing and authentication is by no means restricted only to HfO_x based memristors.

The conductive filament formation in the HfO_x thin-film essentially defines the internal state variable g for a HfO_x

based memristor. The current in the memristor is also determined by the electron tunneling in metal-insulator boundary. Thus, the analytical model of the thin-film evolution in a HfO_x-based memristor can be described using the following equation as [25]:

$$\frac{dg}{dt} = \nu_0 e^{-Ea,m/kT} sinh(\frac{qa\gamma V}{LkT}) \quad (4)$$

where g is the state variable for the device which represents the spatial distance between the conductive filament in the oxide and the metal boundary, q is the electron charge, L is the device filament thickness, V is the applied voltage, T is the device temperature, Ea, m, γ, ν_0 are device dependent physical parameters. The current-voltage relationship in the device is given by the following equation:

$$I(g, V) = I_0 e^{-g/g_0} sinh(\frac{V}{V_0}) \quad (5)$$

where I_0, V_0, g_0 are device dependent physical parameters.

Memristors are usually fabricated in a crossbar fashion. In a crossbar memory array, a memristor unit can contain a single resistor (1M configuration) or a transistor and a memristor (1T1M configuration). In this work, we consider transistor controlled operations because such designs can provide better control, have immunity to sneak path currents, and can deliver reliable performance [1]. Fig. 1. shows basic 1T1M configuration.

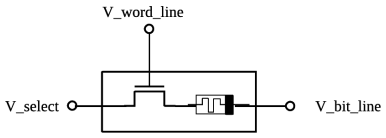

Fig. 1. One-transistor-one-memristor (1T1M) configuration of an memristor cell. A MOSFET is used for controlling the current flowing through the memristor.

B. Standard Features of Memristors

We list intrinsic hardware properties of memristors that are used to design the proposed secret sharing based user authentication protocols.

F1. Non-volatility: Memristors are nonvolatile in nature. For binary operations, two resistance values are associated with each memristor: resistance due to lowest resistive state (LRS) or resistance due to highest resistive state (HRS). However, in analog operation, one finds that there is a continuous spectrum of resistance in between and beyond LRS and HRS that can be achieved. However, for reliable operation, the LRS and HRS of a given device are usually kept near constant and current limiting approach is used for safeguarding against resistance runaway. LRS and HRS are usually different by several orders of magnitude regarding their resistances.

Moreover, the initial states (*i.e.,* the resistive state of an unutilized device) of a given distribution of memristor population can be random. This random distribution along with the non-volatile nature of the memory can be used as a seed for generating secure random keys.

F2. Bias dependent write-time: Write-time of a memristive device can be defined as the time required for switching a device from one resistive state to another (*i.e.,* from LRS to HRS or from HRS to LRS). For example, switching time of the SET/RESET operation of HfO_x devices have been found to be exponentially dependent on the amplitude of the RESET voltage [25] as shown in the Fig 2. Thus, by adjusting the bias voltage, one can manipulate the write-time required for a given state transition. The memristor mentioned above based PUF approaches have utilized this feature [2].

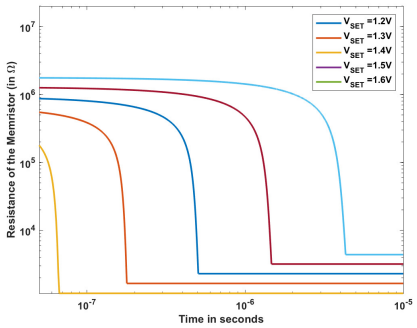

Fig. 2. Variation of memristor SET-time as a function of bias voltage. We have used 1M configuration with $I_0 = 10\mu A$, $L = 12nm$ and the parameters given at [25]. Note that, with a linear increase in the SET voltage, there is an exponential decrease in the total time required for the state-transition

F3. Pulse-Controlled State Transition: Further control and reliability on the filament growth can be achieved by employing pulsed bias voltages, where the duration of the pulse (t_p) is a fraction of the complete write-time (t_{wr}), *i.e.,*

$$t_{wr} = nt_p \quad (6)$$

F4. Fabrication variation in filament thickness: Physical properties of a memristor vary significantly with the manufacturing variation in the filament thickness. For example, equation 4 shows that the change in resistive state is proportional to the $sinh()$ function of the inverse of the film thickness L. Therefore, a small change in film thickness can lead to a significant measurable change in the resistance of the device. Such random variability resulting from fabrication variations can be exploited to design security features such as device authentication. As shown in Fig. 3, the write-time of HRS to LRS transition is longer for memristor with large filament thickness.

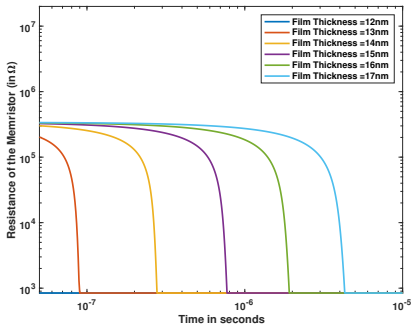

Fig. 3. Variation of memristor SET-time as a function of filament thickness. We have used 1M configuration with $I_0 = 10\mu A$, $V_{SET} = 1.7V$ and the parameters given at [25]. Note that, with a linear increase in the filament thickness, there is an exponential decrease in the total time required for the state-transition

F5. Non-linearity: From the basic equations presented in the previous sections, it can be seen that the resistive-switching devices are highly nonlinear. Device nonlinearity is a highly sought after property for secure hardware design, and therefore, harnessing this property will provide a novel

implementation of standard hardware security protocols such as PUFs. This feature is confirmed in our experiments and can be seen from the figures presented in this section.

C. Key Requirements and Assumptions for Memristor Based Authentication

From the device features described above, below we specify our assumptions on the expected device behavior that are needed for the design of the secret sharing based user authentication protocols, and also give the justification of these requirements and assumptions. For our discussion, let us assume that the SET operation is achieved by a singular or multiple constant duration voltage-pulses which are denoted by ON-pulses.

R1. Changes due to consecutive ON pulses are *additive* in nature.

When given sufficiently high frequency and proper amplitude, short duration ON pulses can cause memristor resistance state transitions just like a longer single voltage pulse. For example, if the device is not already in LRS, two consecutive ON pulses would put the device in a lower resistive state than a single ON pulse. As we can see from equation 4, with V=0, the rate of change of the state is zero, so we would expect a device to retain its resistive state with zero input voltage. Therefore, the device should need the same amount of time with the bias voltage on when it is written using multiple short pulses or a single long pulse. For example, if a device requires a 1.5V-100ns pulse to go from HRS to LRS, it should show similar transition when ten (or a similar number of that order) of 1.5V-10ns pulses are applied. This is validated in our simulations as well.

R2. The memory elements are *monotonic*.

If the device is not in an LRS, applying an ON pulse will always decrease the resistance, and its effect cannot be undone without resetting the device.

R3. Intermediate resistive state transitions are random; however, the number of ON pulses remains nearly the same for different programming cycles.

As we have seen in memristor's nonlinearity feature F5, when the memristor device transitions from one resistive state to another, its intermediate resistance does not change linearly with time. Instead, the change is exponential and has random variations due to local Joule heating and stochastic nature of ionic transportation through the thin-film. The can be verified by the I-V relationship of memristor [1].

R4. The number of short duration pulses during the state transition cannot be predetermined without information about the programming conditions.

This is the discrete version of requirements R1 under the assumption of R3. In another word, one can not imply how many short duration pulses have been applied by observing the intermediate resistance value during the state transition. Furthermore, to learn the exact/an approximate number of pulses required for a given state transition at a given bias, one needs to observe a complete programming cycle. At a given bias, it would be impossible to extrapolate the number of pulses required by measuring the resistance change of the device for an only small period during a transition.

D. Utility Functions

Before proposing memristor based user authentication protocols, we first describe the basic memristor utility functions that are critical for these protocols.

RESET: A RESET operation uses a fast negative biased pulse between the top and the bottom electrode to put a memristor to the high resistive state (HRS).

SET: A SET operation puts a memristor to the low resistive state (LRS). For authentication purpose, we will use the bias dependent write time variability during set (transition from HRS to LRS) operation. We will use multiple short duration ON pulses for setting the device instead of a longer ON pulse.

SET Pulse Count (SPC): The number of ON pulses required to transit a memristor from HRS to LRS.

Pre-condition: A k-bit pre-condition (k<SPC) operation applies k consecutive ON pulses to the memristor device.

Read State: No pulse for one cycle. Then detect whether the memristor device is at HRS or LRS.

IV. User Authentication Protocols

A. Single User Authentication

For single user authentication, a simple solution for secure authentication can be described as follows:

Protocol 1.

Enrollment:

1. For a given memristor R, Alice profiles the complete programming cycle of the memristor with a given (V_select, V_{BL}, V_{WL}). The profiling is done by applying n-consecutive pulses at the word_line with pulse width $t_p = average(t_{wr})/n$ and measuring the corresponding current I_i after i^{th} pulse, where $(1 \leq i \leq n)$.
2. Alice shares this profile (i, I_i) where, $i = 1, \ldots, n$ with Bob.
3. Alice and Bob also shares a random precondition value k.

Authentication:

1. Alice executes the following steps:
   a. **RESET** R.
   b. **Precondition(R,k)**.
   c. Chooses a random number $x \in [1, n-k]$.
   d. **Precondition(R,x)**.
   e. Measures the current I_{rand} with the parameters specified at enrollment step 1.
   f. Send Bob this random current I_{rand} where $I_1 < I_{rand} < I_n$.
2. Using the profile data, Bob returns x', where x' the number of ON pulses required to be applied to a memristor at the k-state to produce I'_{rand}.
3. Alice accepts x' if $abs(X - X') < \epsilon$; where ϵ is a predetermined error threshold.
4. After a successful authentication both updates $k \leftarrow (k+1) \bmod (n-1)$.

This method will provide n-secure authentications in the presence of an eavesdropper. The number of authentication can be increased by using a pseudo-random number generator (PRNG) and update $k \leftarrow PRNG(k) \bmod (n-1)$. Alice can make use of her considerable number of memristors to increase the hardness. However, this mechanism possesses a storage burden to Bob. Furthermore, this protocol requires the extensive capability of current measurement. This problem can be addressed by using differential measurement techniques where the resistance of the memristor is compared with other fixed resistors with different resistances. Another issue with this method is that it requires the device to be ideal, *i.e.,* the device must maintain the similar programming behavior over different cycles. For a reliable application, we need to either reduce such restriction or provide proper error-correction mechanisms for a reliable application.

Next, we present a protocol using the basic schemes of memristor-based secret sharing for single-user authentication. We use a special access structure that computes the XOR of two different physical properties, namely the input voltage and the resistive state of a given memristor. The circuit is shown in Fig 4. was developed by Shin *et. al.* [26] as a resistive computing primitive. The key operating principle of this structure is simple. The resistive state of the output resistance R_P is the XOR of the applied voltage V_X and the input resistance R_Y. Details of the operation of this circuit are discussed in the next section.

Protocol 2.
Enrollment:
1. For a n-bit password, Alice creates an access structure containing n-numbers of three-transistor-two-memristor blocks.
2. Alice and Bob agree on an n-bit random seed ξ and generates $Y \leftarrow PRNG(\xi)$.
3. The i^{th} input resistance R_{Y_i} of each block is configured by Alice using the following rule:
a. If $Y_i = 0$, **RESET** R_{Y_i}.
b. If $Y_i = 1$, **SET** R_{Y_i}.
4. For the i^{th} bit of Bob's password, Alice recodes the response of Bob's password using the following scheme:
a. If the i^{th} bit is 1, Bob choose the i^{th} block of the access structure and applies $V_{xi} = V_{BL}$. Alice stores the resistive state of R_{Pi} as a binary variable H_i (*e.g.,* if $R_{Pi} = LRS$, $H_i = 1$ and if $R_{Pi} = HRS$, $H_i = 0$).
b. If the i^{th} bit is 1, Bob choose the i^{th} block of the access structure and keeps $V_{xi} = 0$ and Alice stores the resistive state of R_{Pi}.
5. After enrollment, the access structure is locked, *i.e.,* the read/write mechanism for all R_{Yi} is disabled.
Authentication:
1. Alice resets all R_p's of the access structure.
2. Bob applies his password at the V_x terminals of Alice's device as described in step 3 of the enrollment phase.
3. Alice records the resistive state of each R_{pi}s and compares with the stored value H_i. If the values match, Alice authenticates Bob.

Note that, this scenario is one of the simplest application of Ito, Saito, and Nishizeki's constructions of secret sharing between two parties [27]. Note that, for each bit of Bob's password, Alice stores only the XORed value of this bit with a random resistance. Therefore, Alice's authentication server is resistant to off-line dictionary attack. To get Bob's password, an attacker not only requires the content of Alice's authentication server but also a knowledge of the access structure used to generate Alice's response.

B. Multiple User Authentication

To address the problems regarding multi-user authentication, we propose to design a Visual Cryptography inspired memristor based multi-user authentication scheme. A small motivational example for 2-out-of-3-user authentication using memristor based threshold detector circuit is given below to illustrate the key concepts.

Assume that an entity Alice wants to simultaneously authenticate three users B1, B2, and B3. Alice chose a random key K, and for each bit of the key, for each user, Alice randomly choose a row from the following matrix C_0 and C_1 using the given rules and gives the corresponding 3-bits to the users. The matrices are given as [13]:

C_0 = all the matrices obtained by permuting the columns of
$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$
C_1 = all the matrices obtained by permuting the columns of
$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$
and the distribution rules are:

i If the i^{th}-bit of the key is 0, distribute the rows of C_0 to B1, B2, and B3.
ii If the i^{th}-bit of the key is 1 distribute the rows of C_1 to B1, B2, and B3.

Note that for a single user, it would be impossible to guess the key bit. For authentication, Alice let any two or more users access the memristor for the duration of three pulses. The users apply their keys to each pulse. The keys are taken through an OR gate as shown in Fig 5. An ON pulse is applied at a cycle if the result of the OR gate is one. Otherwise, no pulse is applied. If the i^{th}-bit of the key is zero, then for only two of the cycles the users will give an ON pulse, but there will be no ON pulse for one cycle. However, if the i^{th}-bit is 1, there will be three ON pulses. As a result, after the duration of three pulses, the device's resistance should be lower if the key was 1 and higher if the key was 0.

A threshold detector can easily detect this and reconstruct the key as shown in Fig 5. This scheme can easily be modified to an n-out-of-n or a k-out-of-n authentication system. One weakness of the above example is that to reconstruct a bit of the key; the users do not necessarily need Alice. If any two of the valid users come together, they can restore the bits. If we want a hardware dependent authentication, where the participating device should also be a factor during authentication, this might cause a collusion problem. However, this issue does not exist if Alice has some interference in key generation.

Protocol 3.
Assumptions:
1. Alice owns the hardware and controls the V_{BL}.
2. Alice knows the number of ON pulses (p_i) required for pushing the resistance of the memristor R_i from a fixed HRS state to the reference resistance (R_{in}).
Initialization: Multi User Key Generation and Registration
1. Alice first choose a random key X and a global bit-line voltage V_{BL}.
2. For each bit $(x_i) \in X$, Alice uses distinct an memristor R_i.
3. Alice considers a ground set G consisting of k elements $g_1, g_2, ...g_k$. The subsets of G with even cardinality are denoted by $e_1, e_2, .., e_{2^{k-1}}$ and odd cardinality are denoted by $q_1, q_2, , q_{2^{k-1}}$.
4. Alice constructs two boolean matrices B_0 and B_1 such that, $B_0[i, j] = 1$ iff $g_i \in e_j$ and $B_1[i, j] = 1$ iff $g_i \in q_j$.
5. Alice constructs collection of C_0 and C_1 by permuting all the column of B_0 and B_1 [13].
6.To share a bit (x_i) of the key X, Alice would follow the rules Q1 and Q2 which are given below:
Q1 If the i^{th}-bit of the key is 0, distribute the rows of C_0 to the users.
Q2 If the i^{th}-bit of the key is 1 distribute the rows of C_1 to the users.

7. Alice also distributes respective ids of the memristor used.

Authentication:

1. For each x_i Alice first RESET the corresponding R_i.
2. Alice would simultaneously accommodate k users to apply their keys in the circuit shown in Fig 5.
3. If the total number of ON pulses applied by the users is correct, corresponding device's resistance should be lower than the reference resistance if the key was 1 or higher if the key was 0 (as designed in the registration phase).
4. Alice can check the generated bit with her secret and simultaneously authenticate k-users.

Due to the fabrication variations, each memristor would have different write-time, which will lead to different (but of the same order) values of p_i for various devices at a given V_{BL}. Since Alice owns the equipment, Alice knows the value of p_i for any given Ri. As different memristor needs different numbers of ON pulses, additional 1s need to be padded in each row of C_0 and C_1. A simpler solution to this problem is to precondition each memristor with this additional 1s. Alice can precondition each memristor so that each requires the same number of additional ON pulses to reach LRS. Another way is to use the block-length defined constructs. Block length is the number of ones resulted by OR-ing all the columns of C_1. Therefore, Alice can easily design block length adjusted matrices C_0' and C_1' depending on the number of ON pulses required for a state transition and follow Q1 and Q2 to share a bit (x_i) of the key X.

To make a hardware dependent authentication scheme, Alice can choose smaller block length or pad less number of 1s at the end of C_1' for some of the random bits of the key X. Since the users do not own the hardware, they do not know the exact value of p_i's for a given R_i. Therefore, Alice can share a 0 by sharing contents of C_1' for cases where C_1' is not correctly generated from C_1. Thus, it would be impossible for k-users (or even all the users) to collude and guess the secret key (X).

To complete the registration phase, Alice would provide each user distinct rows of either C_0' or C_1' (based on Q1 and Q2) for every corresponding bit of X along with the respective id of the memristor used.

C. Protocol Analysis

One of the advantages of our proposed user authentication protocols is that they do not need the computational expensive cryptographic techniques (such as performing the modular exponentiation operation for hundreds of bits). In that sense, our approach can be considered as a lightweight cryptographic method for secret sharing based authentication. Memristor's intrinsic energy efficiency (storing data when power is off) makes this even more attractive.

Adversarial Model For single user authentication, we assume an arbitrarily powerful adversary Malice who is capable of performing eavesdropping and masquerading attack. Malice can claim to be Bob and present false credentials to the authentication service. However, Malice does not know Bob's correct credentials. For multiple user authentications, we assume that Malice has access to only one of the user terminals and her goal is to be authenticated along with other $(k-1)$ users. Based on this attacker model, we now consider the potential attacks to our proposed user authentication protocols and how to defend these attacks again. Since we have three different protocols, we have broken the security analysis on the attack-by-attack basis and discussed Malice's goal and the corresponding countermeasures for each case.

1. Password stealing and dictionary attack For such attacks we assume that Malice knows a dictionary of commonly used passwords or authentication symbols. In Protocol 1, Bob's authentication is not password based, rather it is a challenge-response based scheme, where n secure authentication is possible based on the shared knowledge about the device. Increasing the total number of memristors can sufficiently increase the number of authentication. In this protocol, we keep the memristors response to the input pulse (the state of the memristor p_i) as the secret shared between Alice and Bob. Based on device requirement R1 and R4, the attacker may be able to obtain whether a memristor is at HRS or LRS, but he would not be able to find the cycle-accurate state information of these memristors. Thus he cannot reconstruct the pulse train which, when applied to memristor R_i, will change the current level to I_{rand}.

The second protocol is a password-based protocol; however, the password (ON pulse or no pulse) is stored in memristors of the access structure in the form of their resistive state (HRS or LRS, respectively). Alice only retains the resistive value of the output memristor which permanently stores the hashed value of Bob's password with a random resistance distribution. Therefore, a dictionary-based attack on Alice's database (where Alice's database only contains information about the resistive states of the output resistance R_P will not reveal any information about Bob's password. The only way to retrieve Bob's password would be to probe all the resistance of the access structure. Thus, the second protocol enables two-factor password retrieval on Alice's side. One weakness of the second protocol is this is only 1-time secure. There are two simple solutions for this. First, standard cryptographic methods such as encryption and hashing can be used to securely transmit Bob's password to the V_x terminals.

The third protocol requires collusion of at least k-out-of-n-parties to reconstruct the secret without Alice. Although Alice stores the secret for later authentication, this can be easily prevented by using the second protocol to store the secret in a locked access structure. In such cases, authentication and reconstructing the secret without the access structure will be impossible.

2. Password guessing Malice may launch the exhaustive password guessing attack or some more advanced version based on memristor's special features (for example, applying certain pulse train such as all ON and then observe the states of the memristor devices to reveal password information). This should not be a big concern because it can be easily prevented by disabling the user-name or k-user access after a certain number of consecutive failures.

For k-out-of-n authentication Malice might want to exploit the OR operation performed before applying the aggregated pulse to the memristor. He can try to apply all 1s to gain access along with another $(k-1)$ valid user. However, the attack will fail since the attacker does not know whether the shared secret bit among the users is a 0 or a 1. If he applies a string of 1, it can cause a bit flip for a 0. For example, note the toy example of section IV.B If we modify a row of C_0 to [1 1 1], the OR operation will lead to an erroneous interpretation of 1 at the output and lead to authentication failure for all k-users. This will efficiently create a denial-of-service attack for the users.

3. Denial of service In this attack, Malice attempts to alter the state of the memristor devices that store the password-related information such that when the legitimate user enters the authentic password, the system will not be able to verify.

Due to a lot of memristor devices in the authentication architecture, in single-user authentication it may not be easy for Malice to launch this attack to a particular victim unless she correctly identifies which set of memristor devices correspond to the victim. However, Malice may succeed to attack random victims. To countermeasure such attack, we can restrict user's access to the memristor devices. Unfortunately, this would lead to a denial of service attack in the authentication verifier side, and like other authentication disabling mechanisms, such preventive measure eventually would require intervention from the verifier. Therefore, one of the weaknesses of these protocols comes from DoS attacks.

However, for multiple user authentications providing a train of 1's at user input will create a practical denial of service attack for all k users. A simple finite state machine consisting one D-flip flop and an AND gate can be placed at each user terminal to detect such series of 1s. for k-users the flip-flops can be reset after each cycle along with the memristors.

4. Side channel attacks This is a group of influential attacks where Malice target the vulnerabilities in the hardware implementation of the security primitives and protocols. By measuring the side channel information (such as power, timing, and EM emission) during system's execution, secret information of the system can be revealed.

We are not aware of any side channel attacks to memristor based systems. But should such attacks emerge, most of the existing countermeasures (such as careful engineering, using redundancy, design obfuscation, and so on) should be applicable.

5. Cheating and Secret Reconstruction by User Collusion Apart from the strong adversarial model such as Malice, another form of a weak adversarial model can be assumed from a group of dishonest participants in a k-out-of-n secret sharing model. Such attack is defined as cheating where a group of dishonest users uses fake shares to alter the secret that was distributed at the initial phase. This is a common attack scenario in visual cryptography-based secret sharing, and there are several effective countermeasures proposed by Horng et al. [28]. Since we have used visual cryptography for authentication purposes with the secret shared at the initialization phase, cooperation between multiple dishonest users would create a denial-of-service attack on the authentication system. If Alice uses hardware dependent scheme such as the one discussed at the end of section IV.B and in [29], dishonest users cannot create a cheating attack or reconstruct the secret by themselves since Alice owns the hardware. Additionally, for detecting such attacks, Alice can put redundant information in each share that constructs a 2-out-of-2 secret sharing with Alice and the user.

Finally, the effectiveness of the protocols will also depend on password collision (false negative) and false positive alarms. False negative refers to the case when different passwords are deemed to be authentic for one user. This is a problem of password selection. One standard solution is to add salt as the UNIX password management system does. When users choose their passwords, the system (Alice) can append/combine certain unique data to the user selected pulse train. This will ensure that uniqueness of each user's password. However, the system needs to store the salt data associated with each user. It should be noted that one can try to replicate single-user authentication using PRNGs and known seeds. However, memristors are giving a unique advantage of hardware-dependent hashing for password storage in both cases.

False positive alarms happen when an authentic password is declined. It is more of reliability of the proposed protocols than security breaches. We will elaborate this in the experiment section. Here we point out that environment variations such as temperature, noise, and device aging may all create false positive alarms. We do not expect this to be a serious problem because we do authentication at the cycle-accurate level, not do the matching in the analog domain. Furthermore, like most of the biometric authentication schemes (such as fingerprint or face recognition), it is possible to authenticate the password with a non-perfect match. That is, we do not need to match every pulse in the pulse train to authentic the password.

In summary, when we assume that an outside attacker is only allowed to provide pulse train as the password for authentication and can measure the state changes at a fixed time during the cycles, the attacker can gain very little information, and our user authentication protocols will be stable and secure. However, if the attacker gains physical access to the memristors that we use to store user's password-related information, there will be a good chance for malicious attacks such as denial of service.

V. Hardware Implementation

For developing a memristor-based secret reconstruction and authentication circuit, we need to extract the current state of a given device. It should be noted that improper reading of a memristor cell with high bias voltage across the cell can change its states, and therefore, the read operation should be done carefully. To accomplish this, we have used a simple CMOS compatible differential sense amplifier. The resistance of the device R_x is compared with an on-chip resistance R_{in} that has a threshold resistance value. If $R_x < R_{in}$ the circuit outputs 1, else it outputs a zero. The basic circuit is shown in Fig. 5 which can be used for reconstructing shared secret using Protocol 3.

For Protocol 2, we need a 3T-2M circuit configuration as shown in Fig.4. During the enrollment phase of Protocol 2, the access structure for 1-bit of Bob's password is created by turning on $s1$ and turning off $s2$ and choosing proper V_x, V_p and R_p. During authentication phase, first resetting of R_p is performed by keeping $s1$ off and $s2$ on and $V_p = V_{RESET}$. Then the password evaluation is done by turning on $s1$ and turning off $s2$. Details of the operations are described in section VI. A simple sense amplifier as shown in Fig. 5 can be used to determine the state of the output resistance R_p.

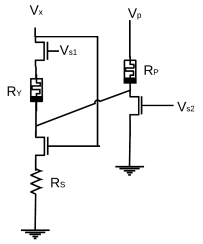

Fig. 4. 3T-2M circuit configuration for single user authentication Protocol 2.

VI. Experiments

We have conducted experiments on the performance of the proposed circuit and protocols and explored their reliability over physical and environmental variation. For these experiments, we have used the Verilog-A model of memristors proposed by Guan et al. [25]. This is a variability-aware RRAM model that takes account of the critical impact of

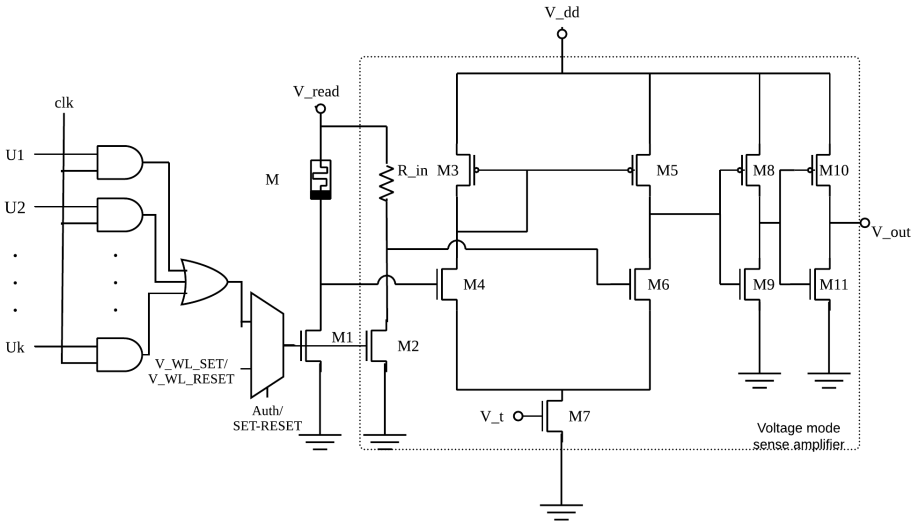

Fig. 5. Circuit for applying the pulsed input from k-users and the assisting voltage-mode sense amplifier-buffer circuit for reconstructing the secret.

TABLE II
OPERATING CONDITIONS USED FOR THE MULTI-USER AUTHENTICATION EXPERIMENTS PRESENTED IN THIS WORK.

Parameter Name	Value
Clock frequency (f_{clk})	50MHz
V_BL_SET	1.8V
V_WL_SET	1.0V
V_WL_RESET	2.7V
V_BL_RESET	1.9V
R_in	250$k\Omega$

temperature change and temporal variations [14]. We have used PTM's 65nm MOSFET models [15] for designing the authentication unit. The simulations are performed in HSPICE platform. For calculations presented in this section, we have used the parameters listed in [25] for the memristor and the experimental conditions are given in Table 2 and 3.

For our first experiment, we provide a simple demonstration of single user authentication using Protocol 2 and the access structure shown in Fig. 4. Here, we assume that the access structure can contain a random resistive state as determined in the registration phase. For this experiment, we consider Bob's password is a binary bit ('1' or '0'). We have demonstrated several consecutive authentication attempts in Fig 6. Note that if Bob's correct password is '0' (or '1'), Alice will see an HRS or LRS in R_P based on the random resistive state of R_Y set at the enrollment phase. Therefore, it would be impossible to reconstruct Bob's actual password by Alice without tampering the access structure.

TABLE I
OPERATING CONDITIONS USED FOR THE SINGLE-USER AUTHENTICATION EXPERIMENTS PRESENTED IN PROTOCOL 2.

Parameter Name	Value
Clock frequency (f_{clk})	20MHz
V_{PRESET}	-1.8V
R_S	10kΩ
V_x	0-2V

Next, we provide a simple 3-out-of-3 secret sharing based user authentication using Protocol 3. For the given parameters,

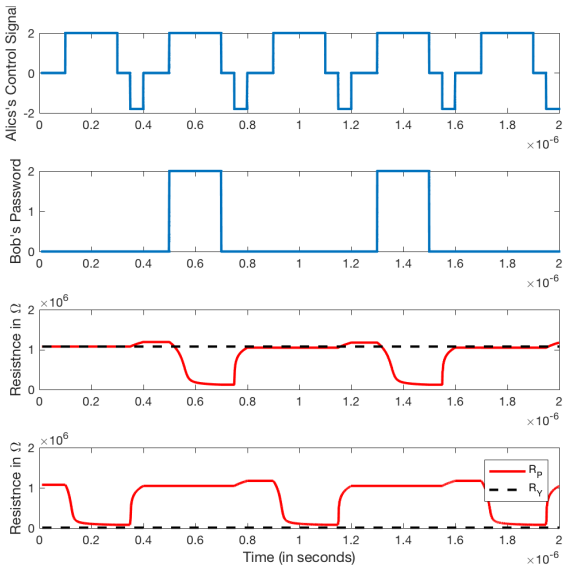

Fig. 6. Simple demonstration of single user authentication. Note that, here Bob's password is "1010" which he applies beginning at 500ns. Before the application of each bit, Alice RESETs the corresponding output memristor R_P. The bottom two figures represent the changes in the resistive state of the output resistance in two different cases. If the configuration resistance in the access structure (*i.e.,* R_Y) is locked at HRS in the enrollment phase (as shown in the third figure from the top) then for Alice would observe a "LRS-HRS-LRS-HRS" pattern in the output resistance R_P, whereas, for R_Y locked in LRS, Alice would see "HRS-LRS-HRS-LRS" pattern. Note that, using one access structure for multiple bits of secret share can reveal information about the secret; therefore, each structure should be used for one-bit of Bob's password.

we find that it requires eight ON pulses (*i.e.,* SET operation) to move the device in question from a high resistive state to the low resistive state. Therefore, after eight consecutive ON pulse to R_x, one would observe logic 1 at V_{out}. Since we want Alice to reveal the secret by applying the final ON pulse, a system with block-length of 7 would be required for doing multi-user

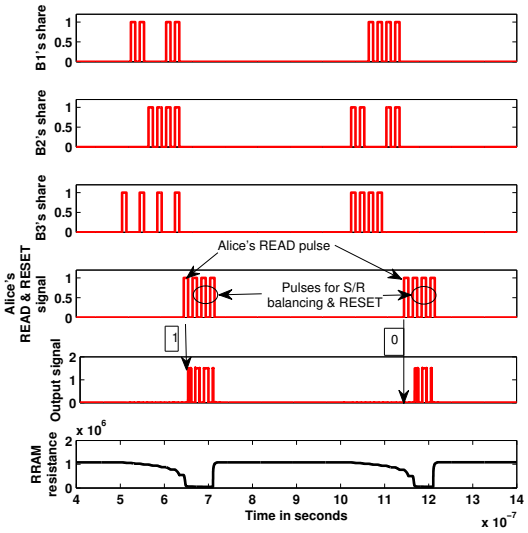

Fig. 7. Simultaneous multi-user authentication using memristor based hardware. Alice verifies a 1 and a 0 shared to the participating entities. The first pulse from Alice is the READ pulse and the three subsequent pulses are for ensuring balanced SET/RESET operation.

authentication. If we consider a 3-out-of-3 user authentication system, then using the constructions provided in [14], we have C_0' and C_1' respectively as:

C_0 = all the matrices obtained by permuting the columns of

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

C_1 = all the matrices obtained by permuting the columns of

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

In Fig. 7, we have shown how Alice verifies a 1 or 0 shared to the participating entities.

VII. Discussions

For the protocols discussed in section V, any of the generated response of the memristive authenticator is entirely dependent on the choice of bias-voltage and the hardware used. Therefore, the reliability of the protocols will be reliant on the reliable performance of the authentication unit over time. Here, we have analyzed the common causes for which an authentication unit will fail to produce the correct response. It should be noted that these weaknesses are mostly due to the imperfect operating conditions of memristor and by addressing these issues, one can develop reliable authentication mechanisms for resistive memory platforms. Primary concerns for the reliability of the proposed protocols are listed below:

1. Unbalanced Set-Reset: The resistive levels of the memristor devices can change due to unbalanced set-reset pulses over a long period of programming cycles, and hence, one of the major sources of error in the authentication protocols can be an unbalanced set-reset cycle. If the RESET pulse pulls up the resistance more than the previous SET pulses, then the overall resistance of the HRS will drift away with time. An example of this effect is illustrated in Fig.8 where we show that due to unbalanced reset, the resistance of the memristor drifts to HRS over multiple cycles. Note that, for proper operation of the authentication mechanism shown in Fig 6. after authentication Alice still needs to apply two consecutive SET pulses for overcoming SET-RESET unbalance problem.

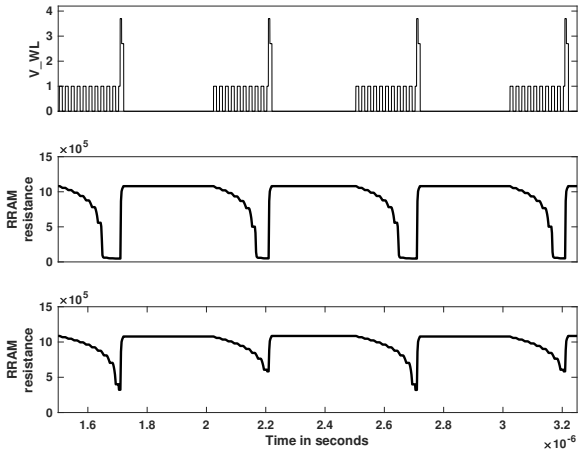

Fig. 8. Example of SET/RESET unbalance for the circuit given in Fig. 5. The top plot shows the typical pulsed programming for a memristive device. The second figure (middle one) lists the changes in a memristor resistance for a stable SET/RESET condition. The plot at the bottom shows the effect of unbalanced biasing where the SET voltage is lowered by 12mV than the previous balanced condition and the RESET voltage is kept the same as before.

To fight error from unbalanced SET/RESET, the authenticator can perform a full SET/RESET operation after every authentication and RESET operation and verify whether the total number of pulses for a SET remains constant. Another error correction technique is, if the authenticator sees the loss/gain of 1 pulse for SET operation after n' operation, then a correction factor 1 can be subtracted/added to the preconditioning or the hashed value. However, after a bit flip is observed, the best solution would be to recalibrate the device using a reference resistance. It can be seen that with fast reset the number of reliable operations before reset correction is small. Therefore, slower reset with longer pulses and a lower bias voltage is preferable to the protocols described here.

2. Temperature dependence of filament formation: Experimental evidence report that the conductive filament formation in a given device is dependent on the device temperature [25]. This can also be seen from equations 4 and 5. A device that goes through balanced SET/RESET is more reliable with the variance in temperature than the one having an unbalanced operation. Therefore maintaining proper operating temperature for the circuit with balanced SET/RESET will be very critical for the reliable operation of the proposed protocols. Changes in the operating temperature will create unreliable operation similar to unbalanced SET-RESET. Therefore, techniques such as a full SET/RESET operation after every authentication, RESET and verify before authentication, and simple error correction as discussed at the previous paragraph *etc.* will be helpful in addressing issues arising from temperature variation.

3. Spatial variations in filament formation and random filament formation: During SET/RESET process lattice temperature of the filament increases abruptly which gives rise to random non-uniform filament generation. This will introduce random fluctuations resistive values during the intermediate states and result in cycle-to-cycle write time variability in the device. To reduce this cycle-to-cycle write-time variability, one can use the same techniques discussed for unbiased SET/RESET.

4. Noise: The primary feature of our proposed designs are to use the bias dependent write-time variability. However, noise in the writing signal can cause cycle-to-cycle variations in the number of pulses required for performing a given write. Noise on the SET or RESET line can profoundly impact the reliability of the circuit, therefore; care should be taken to remove the bit-line noise.

5. Aging: A proper aging model of memristors is still elusive. However, with aging, the response of the circuit would certainly change, and correct aging model and error correction mechanism are required to combat this phenomenon.

VIII. Conclusions

Non-volatile memory based devices and circuits are monotonic. Exploiting this monotonicity can be useful in designing secure circuits and security protocols. In this work, we have connected the additive and monotonic nature of memristor devices with secret sharing based user authentication ideas. We have reported the designed protocol and the necessary circuits required for single and multiple user authentications using memristor based hardware. These robust, hardware dependent user authentication schemes can be useful in developing security primitives in the resource-constrained IoT applications.

Acknowledgment

This work was supported by AFOSR MURI under award number FA9550-14-1-0351. We also thank the reviewers for their valuable comments and feedback.

References

[1] H.-S. P. Wong, H.-Y. Lee, S. Yu, Y.-S. Chen, Y. Wu, P.-S. Chen, B. Lee, F. T. Chen, and M.-J. Tsai, "Metal–oxide RRAM," *Proceedings of the IEEE*, vol. 100, no. 6, pp. 1951–1970, 2012.

[2] G. S. Rose, N. McDonald, L.-K. Yan, and B. Wysocki, "A write-time Based Memristive PUF for Hardware Security Applications," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2013, pp. 830–833.

[3] G. S. Rose, J. Rajendran, N. McDonald, R. Karri, M. Potkonjak, and B. Wysocki, "Hardware Security Strategies Exploiting Nanoelectronic Circuits," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2013, pp. 368–372.

[4] J. Rajendran, R. Karri, and J. B. Wendt, "Nanoelectronic Solutions for Hardware Security."

[5] Y. Wang, W. Wen, H. Li, and M. Hu, "A Novel True Random Number Generator Design Leveraging Emerging Memristor Technology," in *Proceedings of the 25th edition on Great Lakes Symposium on VLSI*. ACM, 2015, pp. 271–276.

[6] J. Sun, Y. Shen, Q. Yin, and C. Xu, "Compound Synchronization of Four Memristor Chaotic Oscillator Systems and Secure Communication," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 23, no. 1, p. 013140, 2013.

[7] B. Muthuswamy and P. P. Kokate, "Memristor-based Chaotic Circuits," *IETE Technical Review*, vol. 26, no. 6, pp. 417–429, 2009.

[8] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC press, 1996.

[9] M. T. Arafin and G. Qu, "RRAM Based Lightweight User Authentication," in *Proceedings of the International Conference on Computer-Aided Design*. IEEE, 2015, pp. 139–145.

[10] A. Shamir, "How to Share a Secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[11] G. R. Blakley, "Safeguarding Cryptographic Keys," *Proc. of the National Computer Conference1979*, vol. 48, pp. 313–317, 1979.

[12] D. R. Stinson, "An Explication of Secret Sharing Schemes," *Designs, Codes and Cryptography*, vol. 2, no. 4, pp. 357–390, 1992.

[13] M. Naor and A. Shamir, "Visual Cryptography," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1994, pp. 1–12.

[14] M. Naor and B. Pinkas, "Visual Authentication and Identification," in *Annual International Cryptology Conference*. Springer, 1997, pp. 322–336.

[15] N. Beckmann and M. Potkonjak, "Hardware-Based Public-key Cryptography with Public Physically Unclonable Functions," in *International Workshop on Information Hiding*. Springer, 2009, pp. 206–220.

[16] R. Maes, *Physically Unclonable Functions*. Springer, 2013.

[17] R. Maes and I. Verbauwhede, "Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions," in *Towards Hardware-Intrinsic Security*. Springer, 2010, pp. 3–37.

[18] G. E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," in *Proceedings of the 44th annual Design Automation Conference*. ACM, 2007, pp. 9–14.

[19] Q. Xia, W. Robinett, M. W. Cumbie, N. Banerjee, T. J. Cardinali, J. J. Yang, W. Wu, X. Li, W. M. Tong, D. B. Strukov *et al.*, "Memristor-CMOS Hybrid Integrated Circuits for Reconfigurable Logic," *Nano letters*, vol. 9, no. 10, pp. 3640–3645, 2009.

[20] K.-H. Kim, S. Gaba, D. Wheeler, J. M. Cruz-Albrecht, T. Hussain, N. Srinivasa, and W. Lu, "A Functional Hybrid Memristor Crossbar-Array/CMOS System for Data Storage and Neuromorphic Applications," *Nano letters*, vol. 12, no. 1, pp. 389–395, 2011.

[21] M. T. Arafin, C. Dunbar, G. Qu, N. McDonald, and L. Yan, "A Survey on Memristor Modeling and Security Applications," in *Sixteenth International Symposium on Quality Electronic Design*. IEEE, 2015, pp. 440–447.

[22] W. Che, J. Plusquellic, and S. Bhunia, "A Non-volatile Memory Based Physically Unclonable Function Without Helper Data," in *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2014, pp. 148–153.

[23] L. Chua, "Memristor-the Missing Circuit Element," *IEEE Transactions on Circuit Theory*, vol. 18, no. 5, pp. 507–519, 1971.

[24] L. Chua and S. M. Kang, "Memristive Devices and Systems," *Proceedings of the IEEE*, vol. 64, no. 2, pp. 209–223, 1976.

[25] X. Guan, S. Yu, and H. P. Wong, "A SPICE Compact Model of Metal Oxide Resistive Switching Memory with Variations," *IEEE electron device letters*, vol. 33, no. 10, pp. 1405–1407, 2012.

[26] S. Shin, K. Kim, and S.-M. Kang, "Memristive XOR for Resistive Multiplier," *Electronics letters*, vol. 48, no. 2, pp. 78–80, 2012.

[27] M. Ito, A. Saito, and T. Nishizeki, "Secret Sharing Scheme Realizing General Access Structure," *Electronics and Communications in Japan*, vol. 72, no. 9, pp. 56–64, 1989.

[28] G. Horng, T. Chen, and D.-S. Tsai, "Cheating in Visual Cryptography," *Designs, Codes and Cryptography*, vol. 38, no. 2, pp. 219–236, 2006.

[29] M. T. Arafin and G. Qu, "Secret Sharing and Multi-user Authentication: From Visual Cryptography to RRAM Circuits," in *Proceedings of the 26th edition on Great Lakes Symposium on VLSI*. ACM, 2016, pp. 169–174.

Md Tanvir Arafin Dr. Md Tanvir Arafin earned his Ph.D. from Department of Electrical and Computer Engineering at the University of Maryland, College Park. His research interests include semiconductor physics, integrated circuits, and embedded security of microelectronic devices.

Gang Qu Dr. Gang Qu has been a professor at the University of Maryland College Park since 2000. He is the director of Maryland Embedded Systems and Hardware Security (MeshSec) lab. His main research interests are in hardware security and trust, low power design, embedded systems, Internet of things, and wireless sensor networks. He has published 200+ papers and delivered 100+ invited talks. He is the author of the first book on VLSI design intellectual property protection (published in 2002). He has taught various security courses at both graduate and undergraduate levels including computer security, trustworthy computing, cybersecurity for the smart grid, hardware security and reverse engineering lab. He has also developed a popular MOOC on hardware security (offered on Coursera since 2015).