Review

Memristors in the Context of Security: A Brief Meta-Review of the State of the Art

Alexander Tekles * , Nico Mexis , Stefan Katzenbeisser

Posted Date: 23 May 2024

doi: 10.20944/preprints202405.1555.v1

Keywords: memristors; security; Physical Unclonable Function (PUF); True Random Number Generator (TRNG)

Article

Memristors in the Context of Security: A Brief Meta-Review of the State of the Art

Alexander Tekles *, Nico Mexis and Stefan Katzenbeisser

Faculty of Computer Science and Mathematics, University of Passau, Innstraße 43, 94032 Passau, Germany; tekles01@ads.uni-passau.de (A.T.); nico.mexis@uni-passau.de (N.M.); stefan.katzenbeisser@uni-passau.de (S.K.)
* Correspondence: tekles01@ads.uni-passau.de

Abstract: Over the last decade, a lot of research has been conducted on memristors. Most of this research focusses on using memristors for Artificial Intelligence (AI) applications and to fabricate non-volatile memory, but also the security aspects of memristors have been examined. The current study summarises and compares five reviews on the security aspects of memristors. These reviews cover two different perspectives: (1) security applications of memristors such as Physical Unclonable Functions (PUFs) or True Random Number Generators (TRNGs), and (2) potential threats when using memristors to train and store neural networks. The comparison of the reviews reveals that different sets of studies are included in the reviews and different characterisations of the studies are provided. This shows that different perspectives are necessary to get a comprehensive overview of the security aspects of memristors. By synthesising the perspectives of different reviews, this study helps to get such an overview.

Keywords: memristors; security; Physical Unclonable Function (PUF); True Random Number Generator (TRNG)

PACS: 89.20.Ff

MSC: 68M25, 68T01, 94C99

1. Introduction

Conventional CMOS (Complementary Metal-Oxide-Semiconductor) hardware has been optimised over the past several decades and has achieved considerable improvements. This has opened up new possibilities, for example in the field of Artificial Intelligence (AI), which requires a lot of computational resources [1]. However, the potential for further optimisation for traditional CMOS technologies is limited. To overcome these limitations, memristors have repeatedly been proposed as possible solutions. Chua [2] was the first to theoretically describe memristors as a basic circuit element in 1971. Although there has been controversy over the characterisation of memristors as a basic circuit element [3], much research has evolved around this technology. Especially since the first implementation of a memristive device in 2008 [4], many studies on memristor characteristics and potential applications have been published.

The main application areas of memristors are AI applications and non-volatile memories [3]. The purpose of using memristors in these application areas is to increase energy efficiency and reduce computation time [1]. A challenge for such applications of memristors is their unreliable behaviour due to inherent randomness [5,6]. However, this randomness has also been used for designing security applications based on memristors. Randomness plays a critical role for several security primitives. Due to their capability to achieve low energy consumption, memristors are especially promising for embedded devices.

In recent years, much research on memristors has been published, some of which was related to security aspects. Some existing reviews summarise parts of this research, but they usually do not cover all relevant aspects or studies. The purpose of this study is to provide an overview of the security aspects of memristors by examining recent reviews on this topic and comparing them with each other. This provides a more holistic perspective than existing reviews. The reviews considered in this study cover two general perspectives on the security aspects of memristors: (1) potential security

 doi:10.20944/preprints202405.1555.v1

applications of memristors like Physical Unclonable Functions (PUFs) or True Random Number Generators (TRNGs), and (2) security threats when using memristors for AI applications and possible countermeasures. Both of these perspectives are considered in this study.

The general outline of the paper is as follows. After providing some background information on memristors and security primitives in Section 2, the reviews included in this study are compared (Section 3) and discussed (Section 4), before drawing some conclusions in Section 5 and proposing future directions for the field in Section 6.

2. Background

Memristors are circuit elements with a variable resistance that depends on the amount of electric charge flowing through a memristor until a certain point in time. Thus, the resistance of a memristor can be modified by applying a voltage bias. By distinguishing different levels of resistance, information can be stored on memristors. Usually, two levels of resistance are distinguished for this purpose, a High Resistance State (HRS) and a Low Resistance State (LRS) (corresponding to `0` and `1` in binary code, respectively).

Various materials have been used to fabricate memristors, but they all have a similar structure consisting of two electrodes with an active layer in-between them. The resistance of the active layer changes when a voltage bias is applied, based on different mechanisms depending on the material used [3]. To efficiently assemble multiple memristors, they are usually arranged in crossbar arrays.

The state of a memristor can be non-volatile or volatile. Non-volatile memristors are especially suitable to be used as memory devices because they are much faster than other non-volatile memory technologies. Volatile memristors switch from HRS to LRS when applying a voltage bias, and spontaneously switch back into HRS after removing the voltage bias. This behaviour allows for the emulation of biological neurones, which also return to their resting state after being stimulated [7]. This makes memristors a promising technology for neuromorphic computing.

Another area of application for memristors is in-memory computation. Performing in-memory computations allows one to bypass the von Neumann bottleneck because no data need to be transferred between the memory and the computing unit. The capability of memristors to perform in-memory computations relies on the fact that data can be stored via resistance levels and the resistance can be modified by applying a voltage bias. Thus, by encoding the data into a voltage pulse (that is, the amplitude and duration) that is applied to a memristor, the data stored on the memristor can be modified [3]. In particular, this allows for the performance of in-memory logic operations or in-memory vector-matrix multiplications [6].

In addition to using memristors for data storage and advanced computation tasks, they have also been proposed for security applications. The two main security applications of memristors are PUFs and TRNGs.

PUFs exploit the intrinsic randomness of hardware devices that arises from small manufacturing variations. A PUF can be queried with an input (challenge) and returns a response that depends on the device and the challenge. Due to the intrinsic randomness of the device, the responses are unpredictable. The desirable characteristics of PUFs are uniqueness (that is, different devices should return different responses for the same challenge), reliability (i.e., a PUF should repeatedly return the same response for the same challenge, even under different environmental conditions), and unclonability (i.e., for a given device, it should be difficult to manufacture another device that returns the same responses) [8].

PUFs can be divided into weak and strong PUFs [8]. Weak PUFs are characterised by high reliability, but they only provide a small number of challenge-response pairs (possibly only one). Thus, the security of a weak PUF depends on not disclosing the response. By contrast, strong PUFs provide a large number of challenge-response pairs such that it is infeasible to enumerate all of these pairs. Therefore, the security of a strong PUF does not depend on keeping responses secret, as long as the challenge-response pairs are only used once.

 doi:10.20944/preprints202405.1555.v1

A specific type of PUFs are Public PUFs (PPUFs) [9]. PPUFs are based on a publicly available simulation model for a PUF device. Running the simulation model takes much longer than using the physical device to generate a response for a given challenge. This allows us to verify that someone has access to the physical device. The verifier can choose a challenge and simulate the response, send the response to the verifying party, and ask for the corresponding challenge. Then it is feasible to test all possible challenges with the physical device, but not with the simulation model.

TRNGs are a security primitive for generating random numbers, which are vital for various cryptographic applications. TRNGs are based on unpredictable physical processes and therefore provide a high degree of randomness, but they are slow in generating random number sequences. In contrast, Pseudo-Random Number Generators (PRNGs) provide an efficient method to generate long sequences of numbers. However, since PRNGs use a deterministic algorithm that depends only on their initial state (seed), their output is not truly random. A typical setup to combine the strengths of PRNGs and TRNGs is to use a TRNG to generate a seed for a PRNG, and then use the PRNG to generate many random numbers.

3. Reviews on the Security Aspects of Memristors

For the following overview and discussion, five review articles [10–14] are considered. These reviews were selected by searching for the most relevant reviews on the security aspects of memristors in Google Scholar. For this purpose, the first five reviews among all scientific works since 2019 have been selected by using the search string "`memristors security`" and sorting the results by relevance (as provided by Google Scholar on October 29, 2023). The search was restricted to works since 2019 to focus on the most recent developments while covering enough time to include relevant reviews. This becomes evident when varying the time span for the search: when considering all works since 2018, the selection of reviews does not change (compared to considering all works since only 2019), while reviews with high citation impact would be excluded when reducing the time period even more.

Regarding the content of the reviews, [10–13] focus on the usage of memristors for security applications. While the reviews of Pang et al. [11], Lv et al. [10], and Singh [13] consider memristors in general, Wang et al. [12] focus on volatile memristors. Zou et al. [14] take a fundamentally different perspective on the security aspects of memristors by describing security threats and possible countermeasures when using memristors for training and storing neural networks. The remainder of this section summarises these reviews and describes the differences between them.

3.1. Memristor-Based PUFs

One major security application of memristors is to use them as PUFs. [10–13] give an overview of studies proposing memristor-based PUF designs. Figure 1 visualises the coverage of studies in the reviews. The figure illustrates that Pang et al. [11] consider a lot more studies than the other reviews. Lv et al. [10] cover considerably more studies than Wang et al. [12] and Singh [13], which both only cover three studies introducing approaches to implement memristor-based PUFs. Whereas Pang et al. [11] and Lv et al. [10] both consider studies that no other of the four reviews covers, all studies considered by Wang et al. [12] or Singh [13] are covered fully by Pang et al. [11] and partially by Lv et al. [10].

 doi:10.20944/preprints202405.1555.v1

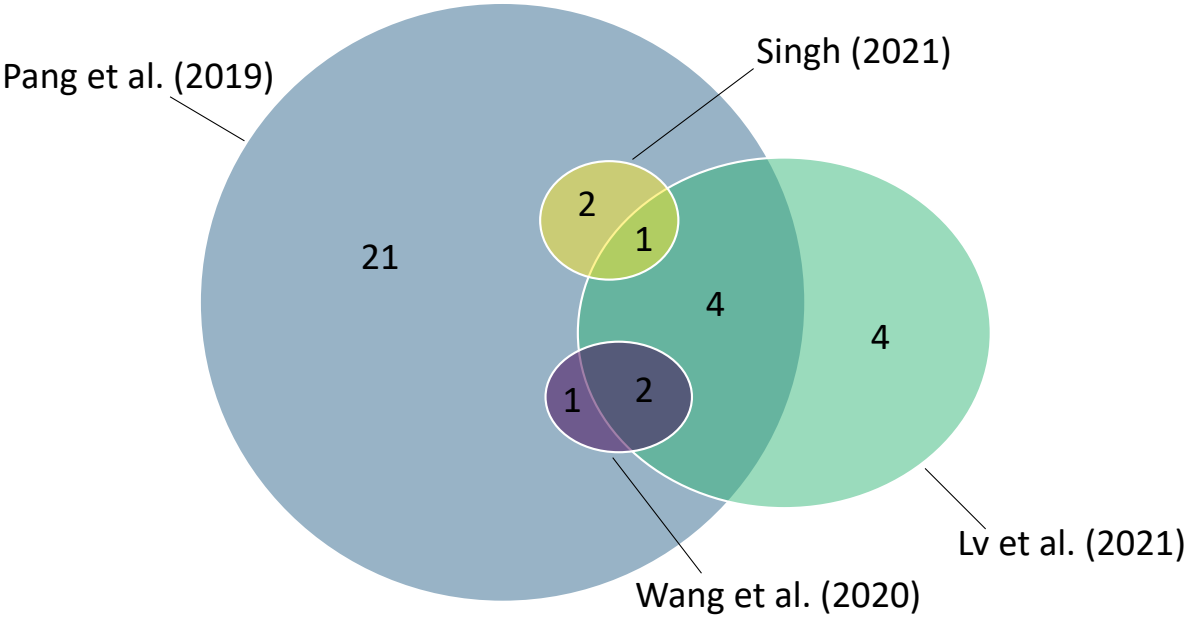

Figure 1. Number of studies introducing a memristor-based PUF approach that are covered in [10–13].

Pang et al. [11] distinguish four types of memristor-based PUFs. Two of these types are weak and strong memristor-based PUFs, following the general distinction between weak and strong PUFs described in Section 2. Weak memristor-based PUFs exploit small variations in the switching probability (i.e. resistance variability) between different memristor cells in a crossbar array. The general idea behind this approach is to apply a voltage bias to memristor cells such that their switching probability is close to 50%. Then, the memristor cells randomly settle to a state, and these states can be used as a sequence of random bits. A straightforward approach to implement strong memristor-based PUFs is to compare the resistance between two memristor cells in a crossbar array. Due to the resistance variability of memristors, this also allows the generation of random bits. In this approach, the position of the two cells to compare can be regarded as a PUF challenge. This yields a large number of possible challenge-response pairs, making this approach a suitable strong PUF design. Another approach to implement memristor-based strong PUFs mentioned in [11] is to use memristor-based arbiter PUFs or memristor-based ring oscillator PUFs.

Pang et al. [11] identify, in addition to the distinction between weak and strong memristor-based PUFs, two further types of memristor-based PUFs corresponding to specific use cases. The first of these use cases are memristor-based PPUFs. As described in Section 2, a publicly available simulation model of a PPUF device (here: a memristor crossbar) serves as a verification tool. The simulation model for a memristor crossbar is based on the physical characteristics of its memristor cells, and simulation requires computationally heavy operations. Simulating the output for a large memristor crossbar is infeasible because this would require the simulation of all possible current paths, which grow exponentially with the crossbar's size. However, simulating the output of a subset of memristor cells is feasible. In contrast to the simulation model, the corresponding physical device can be used to compute the output of the whole memristor crossbar in a reasonable amount of time. This discrepancy in the runtime between the simulation model and the physical device can be exploited to define an authentication protocol [15].

The last type of memristor-based PUFs identified by Pang et al. [11] is characterised by its strong resistance to machine learning attacks. Machine learning attacks can be a threat to PUFs if there are strong correlations among the PUF's challenge-response pairs. The proposed design uses two layers of memristor cells. The challenge specifies a set of cells in the first layer, and the (random) resistance states of these cells in the first layer are used to select one cell in the second layer. The output of this cell in the second layer is then used as the PUF's response bit. Due to the increased complexity, this

 doi:10.20944/preprints202405.1555.v1

approach is better suited to prevent machine learning attacks than a PUF design based on a single memristor layer or a conventional arbiter PUF.

Similar to Pang et al. [11], Lv et al. [10] also identify nano-PPUFs as one type of memristor-based PUFs. However, in addition to this type of memristor-based PUFs, Lv et al. [10] propose a different categorisation compared to Pang et al. [11]. One of the corresponding types of memristor-based PUFs that Lv et al. [10] refer to are hybrid memristor-CMOS PUF circuits. In this setup, memristor cells are combined with ordinary CMOS-based PUF designs like arbiter PUFs or ring oscillator PUFs. Because the structure of memristors is compatible with the structure of CMOS hardware, they can be efficiently integrated into one device. Combining memristors and CMOS hardware in one device can not only increase a PUF's security, but also allows to reduce hardware resources and avoid costly post-processing compared to PUFs solely based on CMOS hardware. Furthermore, Lv et al. [10] identify a PUF design based on a diffusive (volatile) memristor crossbar as another type of memristor-based PUF. For this PUF design, Lv et al. [10] refer to a study that is categorised as a weak memristor-based PUF approach in [11].

Finally, Lv et al. [10] describe a memristor-based approach to prove the destruction of cryptographic keys stored on a device. Such a functionality allows one to revoke a key on a remote device or restrict its validity. Proving the destruction of keys on CMOS-based devices is difficult due to their volatile nature, whereas the non-volatile storage capability of memristors can be exploited for such a task. The idea of this approach is to use the resistance variability in the LRS among the cells of a memristor crossbar array to generate a fingerprint of a device. The same crossbar array is used to store a cryptographic key. Then, the fingerprint can only be extracted if no data (i.e., no cryptographic key) is stored on the crossbar array because only in this case can the resistance in the LRS be measured for all the cells. Since the variability in the LRS depends on random manufacturing variations, this approach can be considered a PUF design.

Wang et al. [12] focus on volatile memristors and describe only one study introducing a PUF design. This study is also included in [11] (referred to as a weak PUF) and [10] (referred to as a diffusive memristor-based PUF). Besides this study, Wang et al. [12] mention two other studies that are only cited as exemplary PUF designs based on non-volatile memristors without describing their approaches. Singh [13] also only considers three studies for an overview of memristor-based PUFs. Based on these three studies, Singh [13] identifies two types of memristor-based PUF designs. First, a design based on a memristor crossbar array is sketched that follows the idea of weak memristor-based PUFs described in [11]. Second, Singh [13] identified a memristor-based PPUF design, similar to and based on the same study as in [10,11].

3.2. Memristor-Based TRNGs

The second major security application of memristors is memristor-based TRNGs. Similarly to memristor-based PUFs, [10–13] give an overview of memristor-based TRNG approaches. Again, Pang et al. [11] cover more studies than the other reviews (see Figure 2). The reviews generally cover fewer studies proposing memristor-based TRNG designs than studies proposing memristor-based PUF designs. In contrast to the studies on memristor-based PUFs, there is less overlap between the reviews on studies on memristor-based TRNGs. For example, each review considers at least one study that is not considered by any of the other reviews.

 doi:10.20944/preprints202405.1555.v1

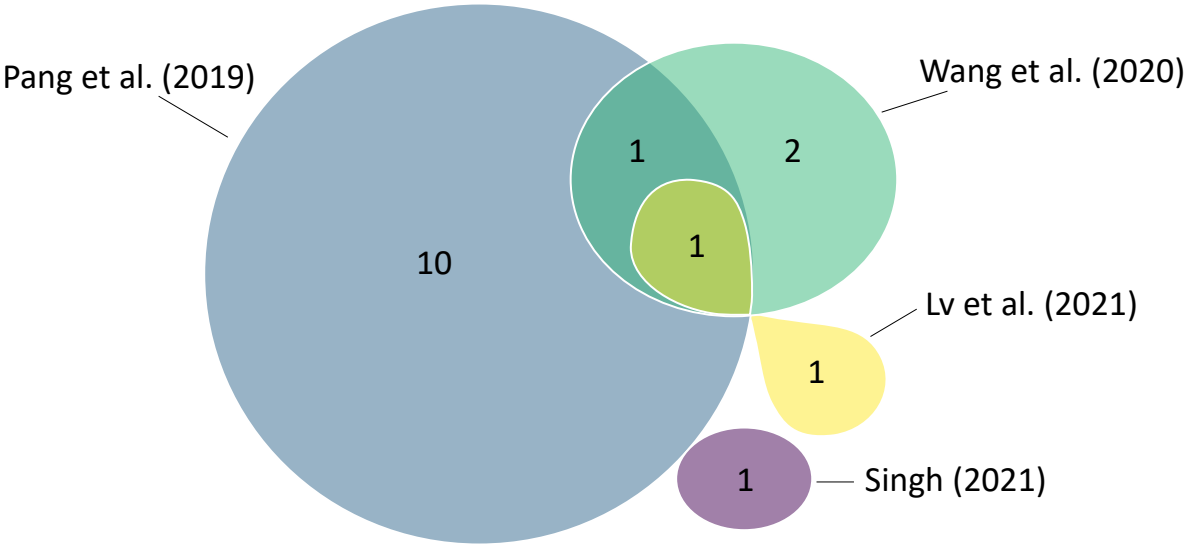

Figure 2. Number of studies introducing a memristor-based TRNG approach that are covered in [10–13].

A straightforward approach to implement memristor-based TRNGs described by Pang et al. [11] and Lv et al. [10] is to exploit the probabilistic switching behaviour of memristors. When applying a voltage bias, the cumulative probability function of cells switching between HRS and LRS follows a lognormal distribution. The shape of this distribution depends on the amplitude and duration of the voltage bias. This can be used to generate random bit patterns with a memristor crossbar array by choosing an adequate voltage bias amplitude and duration (e.g., implying a 50% switching probability).

Another mechanism that allows the implementation of memristor-based TRNGs described by Pang et al. [11] is Random Telegraph Noise (RTN). RTN occurs due to the random trapping and de-trapping of charge carriers, leading to sudden unpredictable changes in current levels [3]. Compared to CMOS-based TRNGs, implementations exploiting the memristors' RTN require less power and can be implemented without a preamplifier. However, RTN-based TRNGs can only achieve a low output frequency. Another drawback is that the RTN frequency and amplitude are difficult to control, which may lead to instabilities in the random number output and require post-processing of the output.

The next approach to implement memristor-based TRNGs described by Pang et al. [11] is to use current fluctuations in a memristor cell (while the cell is settled in either HRS or LRS). More specifically, current differentials over time can be modelled to follow a Gaussian distribution. This pattern is attributed to Brownian motion, which is a random physical process. Thus, current differences are a good source of randomness for TRNGs [16]. Although using current differences to implement memristor-based TRNGs results in a more stable output and is easier to control than RTN-based TRNGs, Pang et al. [11] identify low output frequency as a disadvantage of this approach.

Pang et al. [11] also describe a memristor-based TRNG design that provides a higher output frequency than the aforementioned approaches. In this design, resistance variations of memristor cells across set-reset cycles are used as a source of randomness. To achieve a high output frequency based on cycle-to-cycle resistance variation, a memristor with high switching speed and high endurance is necessary. Another limitation of this approach is its high power consumption due to the high switching frequency.

The last type of memristor-based TRNGs described by Pang et al. [11] uses the write delay time of a volatile memristor as a source of randomness. In particular, the delay time between a voltage pulse and the following switch of a volatile memristor from HRS to LRS is used. Since this delay time follows a random distribution, it can be used to implement a TRNG. Besides good randomness and stability, the main advantage of this approach compared to CMOS-based TRNGs and the aforementioned

 doi:10.20944/preprints202405.1555.v1

memristor-based TRNGs is that no post-processing is necessary. Since this design also requires frequent switches between HRS and LRS, Pang et al. [11] mention challenges similar to those for TRNGs based on cycle-to-cycle resistance variation: the memristor must have a high endurance and high switching speed, and high power consumption due to repeated switching.

For describing the approach to using the write delay time of a volatile memristor to implement a TRNG, Pang et al. [11] refer to one particular study. This study is also considered in the reviews of Lv et al. [10] and Wang et al. [12]. Whereas Pang et al. [11] mention high power consumption as a challenge of this TRNG, Lv et al. [10] and Wang et al. [12] argue that it has low power consumption compared to other TRNGs (CMOS-based TRNGs, TRNGs based on non-volatile memristors, and further TRNGs based on volatile memristors). The viewpoint of Lv et al. [10] and Wang et al. [12] is also in line with the original study proposing this TRNG implementation [17]. However, Pang et al. [11] focus on comparing the approach to other types of memristor-based TRNGs described in their review, some of which do not require to repeatedly switch the memristor cells and therefore consume less power than the TRNG implementation of Jiang et al. [17]. Thus, the different perspectives on the role of power consumption in the design proposed by Jiang et al. [17] is a matter of reference approaches considered for comparison rather than differences in the evaluation of the approach itself.

In addition to the TRNG design proposed by Jiang et al. [17], Wang et al. [12] also include a second study that proposes a TRNG design based on the write delay time in volatile memristors in their review. This implementation achieves a smaller overall circuit area compared to the design proposed by Jiang et al. [17] by replacing a comparator and a resistor with a second memristor and an AND gate. However, the operation voltage of this design is higher than for the approach of Jiang et al. [17], leading to a higher power consumption when generating random numbers [12]. Despite explicitly focussing on volatile memristors, Wang et al. [12] also consider TRNG designs based on non-volatile memristors. In particular, approaches based on RTN and probabilistic switching of memristors are described.

The review of Singh [13] considers only one study that proposes a memristor-based TRNG design. In this design, memristors are included in ring oscillators, which increases the entropy compared to purely CMOS-based ring oscillators.

3.3. Further Security Applications of Memristors

Lv et al. [10] and Singh [13] mention further security applications of memristors in addition to PUFs and TRNGs in their reviews. However, only short and high-level descriptions are provided, which is why this section only gives a brief overview of these applications.

One security application mentioned by both Lv et al. [10] and Singh [13] is to use memristors to implement chaotic circuits. Chaotic circuits are deterministic nonlinear systems that generate an unpredictable signal that can be used to encrypt messages. Memristors are suitable components for chaotic circuits because of their nonlinear behaviour. Other security applications of memristors mentioned by Singh [13] are using them for tamper detection, forensics, or efficient encryption techniques. Tamper detection and forensics may be facilitated by the phenomenon that the resistance of memristors changes slightly as a result of read operations. This reduction in resistance allows one to detect unauthorised access to data stored on a memristor. Efficient encryption of data stored on memristors can be achieved based on the randomness of sneak paths in a memristor crossbar array.

3.4. Security Threats of Using Memristors in AI

Zou et al. [14] are not concerned with security applications of memristors in their review, but with security threats when using memristors for training and storing neural networks. Memristors can be a valuable technology for these tasks due to their capability to perform in-memory computations and their potential non-volatility. However, using memristors to train and store neural networks may also bring about security threats. When discussing such threats and potential countermeasures, Zou et al. [14] distinguish between black-box and white-box attack models. The black-box attack model

 doi:10.20944/preprints202405.1555.v1

assumes that the attacker can manipulate inputs, observe outputs, and observe side channels. In the white-box attack model, the attacker also has access to the trained weights of a neural network model.

One type of black-box attacks considered by Zou et al. [14] are learning attacks. The goal of learning attacks is to steal a proprietary neural network[1] that is not openly available itself, but for which input-output pairs can be collected (e.g., by querying the model with selected inputs). The input-output pairs can then be used to train a new neural network, which predicts the predictions of the first model. If an attacker has physical access to a device that stores the weights of a neural network, memristors can help prevent learning attacks. For certain memristors, their resistance changes over time or with every read operation. Thus, data stored on such memristors need to be refreshed periodically or after some read operations. If a neural network is stored on such a device and refreshing the memristor cells is only allowed for authorised users, an unauthorised user can only collect a limited number of input-output pairs. This can prevent stealing the model stored on the device because sufficient training data are necessary to achieve good performance of a neural network.

The second type of black-box attacks described by Zou et al. [14] are side-channel attacks. Side-channel attacks exploit the side effects of a device during operation, such as power consumption or runtime, to extract confidential information. Memory access patterns have been described as a possible target for side-channel attacks to gain information about the structure of a neural network [18]. Zou et al. [14] argue that this attack may also be possible for neural networks stored on memristive devices. A countermeasure against such attacks are oblivious RAM algorithms, which hide memory access patterns [14].

Using a white-box attack model implies more security threats for neural networks stored on memristive devices than a black-box attack model. Since memristors can be non-volatile, weights can be stored permanently on memristive devices. Thus, if an attacker has physical access to a memristive device storing the weights of a neural network, the attacker may be able to simply read them or use micro-probing techniques [14].

The general strategy to mitigate this threat is to modify the weights of a neural network so that no inference is anymore possible without additional information. A straightforward way to achieve such an obfuscation of neural network weights stored on a memristive device is to encrypt each weight. However, this approach adds considerable overhead due to repeated encryption and decryption operations during inference. This overhead can be reduced by only encrypting the most significant weight of each layer of the neural network. Without the most significant weight in each layer, no reasonable inference is possible anymore [14]. Another approach to protecting the weights of a neural network is to permute them. The permuted weights allow efficient inference if the permutation applied to the original weights is known. Otherwise, the permuted weights do not reveal enough information to perform inference operations.

To bind a neural network to a specific device, the aforementioned techniques to obfuscate weights can be combined with fingerprinting the device. The randomness inherent in memristors can be used for this purpose, similar to memristor-based PUFs or TRNGs.

4. Discussion

Even though [10–13] all deal with security applications of memristors, they cover different sets of studies. Pang et al. [11] generally include considerably more studies than the other reviews. Likewise, Lv et al. [10] cover considerably more studies on memristor-based PUF approaches than Wang et al. [12] and Singh [13], but these three reviews differ only slightly with regard to the coverage of studies on memristor-based TRNG approaches. Thus, Pang et al. [11] provide the most comprehensive overview

[1] Learning attacks can be applied to any type of machine learning model that calculates predictions for given inputs. Here, only neural networks are considered because Zou et al. [14] also focus on neural networks.

 doi:10.20944/preprints202405.1555.v1

of memristor-based PUFs and TRNGs among the reviews examined. The relatively small number of studies included in Wang et al. [12] can be attributed to their limited focus on volatile memristors.

Lv et al. [10] and Singh [13] provide a wider perspective on security applications than Pang et al. [11] by mentioning additional applications in addition to PUFs and TRNGs. However, the reviews of Lv et al. [10] and Singh [13] contain shorter and rather high-level descriptions compared to Pang et al. [11], which makes it difficult to get a complete picture of the security applications of memristors.

Pang et al. [11] and Lv et al. [10] both provide a categorisation of memristor-based PUFs. However, the two reviews refer to different categories for this purpose. Pang et al. [11] consider potential use cases for their categorisation: the distinction between weak and strong memristor-based PUFs is rather general, whereas memristor-based PPUFs and the PUF design aimed at defending against machine learning attacks are rather specific use cases. The two specific PUF approaches can also be regarded as strong PUFs because they allow to generate a large number of challenge-response pairs.

Lv et al. [10] consider not only potential use cases (PPUFs, proving key destruction), but also particular characteristics of the memristive devices (e.g., hybrid memristor-CMOS structure vs. memristor crossbar arrays, volatile vs. non-volatile memristors) for their categorisation of memristor-based PUFs. While a categorisation based on potential use cases allows one to select the right approach for a given problem, referring also to the characteristics of memristors gives a better overview of existing techniques to construct memristor-based PUFs.

The memristor-based TRNGs described by Pang et al. [11] are categorised according to the sources of randomness on which the TRNG designs are based. The other reviews do not provide such a categorisation of memristor-based TRNGs, but only describe the small number of studies they include individually without grouping them into generalised categories.

The fact that security applications of memristors apart from PUFs and TRNGs (e.g., chaotic circuits) are either not even mentioned in the reviews or only very briefly discussed suggests that such applications do not play an important role. However, if a review aims to give a comprehensive overview of the security applications of memristors, all applications discussed in the literature should be considered. Alternatively, if reviews focus on certain applications, this restriction should at least be explicitly stated.

The review of Zou et al. [14] shows that the usage of memristors to learn and store neural networks may also have security implications. Some security threats and countermeasures mentioned in this review specifically apply to memristors (e.g., side-channel attacks on memristors, or using resistance drift in memristors to thwart learning attacks). However, other threats and countermeasures are relevant independently of the underlying hardware. For example, learning attacks or encrypting the weights of a model can be performed on memristive devices as well as other devices. This distinction between techniques that specifically target memristive devices and hardware-agnostic techniques is somewhat unclear in [14], making it difficult to assess the security implications of memristors in artificial intelligence systems. Nevertheless, reviewing security threats and countermeasures when using memristors to learn and store neural networks adds an important perspective on the security aspects of memristors.

5. Conclusion

During the last decade, various studies have addressed the security aspects of memristors, and several reviews have synthesised parts of this research. The current study summarises and compares five of these reviews that have recently been published. The differences among the reviews show that each of them has a different focus. With regard to security applications of memristors, Pang et al. [11] provide the most extensive overview of memristor-based PUFs and TRNGs, whereas Lv et al. [10] and Singh [13] provide a wider perspective by considering further security applications in addition to PUFs and TRNGs. Wang et al. [12] have a narrow focus on volatile memristors, but are also concerned with security applications such as PUFs and TRNGs. By contrast, Zou et al. [14] contribute a very

 doi:10.20944/preprints202405.1555.v1

different perspective by addressing security threats and countermeasures when using memristors for training and storing neural networks.

6. Future Directions

The current paper gives an overview of existing research on memristors and security, but also reveals potential directions for future research. For example, new reviews on the security aspects of memristors could contribute to a more comprehensive overview of this topic. All reviews covered here have been published in 2021 or earlier, which means that no primary literature published within the last two years is considered. Reviewing the most recent research on security aspects of memristors could fill this gap. Another potential contribution of future reviews could be a more standardised review approach compared to the reviews covered here. This may help to increase the comparability of security applications by unambiguously highlighting the advantages and disadvantages of certain approaches. For example, this could help avoid inconsistencies like the one described in Section 3.2, where a TRNG design based on a volatile memristor is characterised by having a high power consumption by Pang et al. [11] and as having a low power consumption by Lv et al. [10] and Wang et al. [12]. Furthermore, future reviews could analyse how different materials and structures facilitate security applications. These physical characteristics of memristors have not been addressed in the current paper, and the reviews covered here only provide general descriptions of possible memristor setups without relating them to security applications.

In addition to reviews for a better overview of existing studies on security-related topics of memristors, future research may investigate how to use memristors for security applications. The reviews considered here show that by far the most research on security aspects of memristors focusses on designing memristor-based PUFs and TRNGs. However, other applications are also possible, some of which are mentioned by Lv et al. [10] and Singh [13]. Thus, more research on such applications (e.g., memristor-based chaotic circuits, tamper detection, forensics, efficient encryption) apart from memristor-based PUFs and TRNGs may be promising directions for future research. To use memristors for security applications in practice, future research could also help to understand how memristors and CMOS hardware can be integrated in one device. This would allow to implement memristor-based security applications at scale. Each TRNG implementation should also be tested with both the NIST SP800-22 Statistical Test Suite [19] and the NIST SP800-90B Entropy Source Test Suite [20] to ensure that their output is truly random. Analogous metrics should also be used to ensure robustness, unpredictability, and unclonability of memristor-based PUF implementations, such as the Hamming weight, Shannon entropy and intra-device and inter-device Hamming distances, and Jaccard indices, or even spatial auto-correlation and cross-correlation [21].

The current paper shows that meta-reviews can help to get a comprehensive overview of a certain topic. This is especially true for research related to memristors, as memristors are a hot topic with many recently published studies. In particular, many papers on the use of memristors for AI applications have been published over the last few years, including reviews of research in this field. These reviews provide an overview of some research on memristors and AI, but usually focus on a particular perspective. Some existing reviews focus on certain applications of memristors in the context of AI-like approaches to mimic characteristics of biological neurones (e.g. synaptic plasticity) [22] or to reduce energy consumption and latency when training AI models using in memory computation and parallel multiply accumulate operations [23]. Other reviews are primarily concerned, for example, with a comparison of CMOS and memristor technologies for neuromorphic computing [24], hybrid memristor-CMOS designs for training neural systems [25], or comparisons of memristors based on the materials used [26]. Meta-reviews can help to synthesise these different perspectives on memristors in the context of AI.

Author Contributions: Conceptualization, A.T.; methodology, A.T.; validation, A.T. and N.M.; investigation, A.T. and N.M.; data curation, A.T.; writing—original draft preparation, A.T. and N.M.; writing—review and editing, A.T., N.M. and S.K.; visualization, A.T. and N.M.; supervision, S.K.; project administration, S.K.; funding acquisition, S.K.

 doi:10.20944/preprints202405.1555.v1

Funding: This research was funded, as part of Project BA0100016: "CySeReS-KMU: Cyber Security and Resilience in Supply Chains with focus on SMEs", by the Interreg VI-A Programme Germany/Bavaria–Austria 2021–2027 – Programm INTERREG VI-A Bayern–Österreich 2021–2027, which is co-funded by the European Union.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
CMOS	Complementary Metal-Oxide-Semiconductor
HRS	High Resistance State
LRS	Low Resistance State
PPUF	Public Physical Unclonable Function
PRNG	PseudoRandom Number Generator
PUF	Physical Unclonable Function
RTN	Random Telegraph Noise
TRNG	True Random Number Generator

References

1. Mehonic, A.; Sebastian, A.; Rajendran, B.; Simeone, O.; Vasilaki, E.; Kenyon, A.J. Memristors—from in-memory computing, deep learning acceleration, and spiking neural networks to the future of neuromorphic and bio-inspired computing. *Advanced Intelligent Systems* **2020**, *2*, 2000085. doi:10.1002/aisy.202000085.
2. Chua, L. Memristor—The missing circuit element. *IEEE Transactions on Circuit Theory* **1971**, *18*, 507–519. doi:10.1109/TCT.1971.1083337.
3. Lanza, M.; Sebastian, A.; Lu, W.D.; Le Gallo, M.; Chang, M.F.; Akinwande, D.; Puglisi, F.M.; Alshareef, H.N.; Liu, M.; Roldan, J.B. Memristive technologies for data storage, computation, encryption, and radio-frequency communication. *Science* **2022**, *376*, eabj9979. doi:10.1126/science.abj9979.
4. Strukov, D.B.; Snider, G.S.; Stewart, D.R.; Williams, R.S. The missing memristor found. *Nature* **2008**, *453*, 80–83. doi:10.1038/nature06932.
5. Park, S.O.; Jeong, H.; Park, J.; Bae, J.; Choi, S. Experimental demonstration of highly reliable dynamic memristor for artificial neuron and neuromorphic computing. *Nature Communications* **2022**, *13*, 2888. doi:10.1038/s41467-022-30539-6.
6. Yang, X.; Taylor, B.; Wu, A.; Chen, Y.; Chua, L.O. Research progress on memristor: From synapses to computing systems. *IEEE Transactions on Circuits and Systems I: Regular Papers* **2022**, *69*, 1845–1857. doi:10.1109/TCSI.2022.3159153.
7. Wang, Z.; Joshi, S.; Savel'ev, S.; Song, W.; Midya, R.; Li, Y.; Rao, M.; Yan, P.; Asapu, S.; Zhuo, Y.; Jiang, H.; Lin, P.; Li, C.; Yoon, J.H.; Upadhyay, N.K.; Zhang, J.; Hu, M.; Strachan, J.P.; Barnell, M.; Wu, Q.; Wu, H.; Williams, R.S.; Xia, Q.; Yang, J.J. Fully memristive neural networks for pattern classification with unsupervised learning. *Nature Electronics* **2018**, *1*, 137–145. doi:10.1038/s41928-018-0023-2.
8. Herder, C.; Yu, M.D.; Koushanfar, F.; Devadas, S. Physical unclonable functions and applications: A tutorial. *Proceedings of the IEEE* **2014**, *102*, 1126–1141. doi:10.1109/JPROC.2014.2320516.
9. Beckmann, N.; Potkonjak, M. Hardware-based public-key cryptography with public physically unclonable functions. Information Hiding; Katzenbeisser, S.; Sadeghi, A.R., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg, 2009; pp. 206–220. doi:10.1007/978-3-642-04431-1_15.
10. Lv, S.; Liu, J.; Geng, Z. Application of memristors in hardware security: A current state-of-the-art technology. *Advanced Intelligent Systems* **2021**, *3*, 2000127. doi:10.1002/aisy.202000127.
11. Pang, Y.; Gao, B.; Lin, B.; Qian, H.; Wu, H. Memristors for hardware security applications. *Advanced Electronic Materials* **2019**, *5*, 1800872. doi:10.1002/aelm.201800872.

12. Wang, R.; Yang, J.Q.; Mao, J.Y.; Wang, Z.P.; Wu, S.; Zhou, M.; Chen, T.; Zhou, Y.; Han, S.T. Recent advances of volatile memristors: Devices, mechanisms, and applications. *Advanced Intelligent Systems* **2020**, *2*, 2000055. doi:10.1002/aisy.202000055.
13. Singh, J. Implementation of memristor towards better hardware/software security design. *Transactions on Electrical and Electronic Materials* **2021**, *22*, 10–22. doi:10.1007/s42341-020-00269-x.
14. Zou, M.; Du, N.; Kvatinsky, S. Review of security techniques for memristor computing systems. *Frontiers in Electronic Materials* **2022**, *2*. doi:10.3389/femat.2022.1010613.
15. Rajendran, J.; Rose, G.S.; Karri, R.; Potkonjak, M. Nano-PPUF: A memristor-based security primitive. 2012 IEEE Computer Society Annual Symposium on VLSI. IEEE, 2012, pp. 84–87. doi:10.1109/ISVLSI.2012.40.
16. Wei, Z.; Katoh, Y.; Ogasahara, S.; Yoshimoto, Y.; Kawai, K.; Ikeda, Y.; Eriguchi, K.; Ohmori, K.; Yoneda, S. True random number generator using current difference based on a fractional stochastic model in 40-nm embedded ReRAM. 2016 IEEE International Electron Devices Meeting (IEDM). IEEE, 2016, pp. 4.8.1–4.8.4. doi:10.1109/IEDM.2016.7838349.
17. Jiang, H.; Belkin, D.; Savel'ev, S.E.; Lin, S.; Wang, Z.; Li, Y.; Joshi, S.; Midya, R.; Li, C.; Rao, M.; others. A novel true random number generator based on a stochastic diffusive memristor. *Nature Communications* **2017**, *8*, 882. doi:10.1038/s41467-017-00869-x.
18. Hua, W.; Zhang, Z.; Suh, G.E. Reverse engineering convolutional neural networks through side-channel information leaks. Proceedings of the 55th Annual Design Automation Conference; Association for Computing Machinery: New York, NY, USA, 2018; DAC '18, pp. 1–6. doi:10.1145/3195970.3196105.
19. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E.; Leigh, S.; Levenson, M.; Vangel, M.; Banks, D.; Heckert, A.; Dray, J.; Vo, S.; Bassham III, L.E. A statistical test suite for random and pseudorandom number generators for cryptographic applications. National Institute of Standards and Technology (NIST) Special Publication 800-22, Revision 1a, 2010. doi:10.6028/NIST.SP.800-22r1a.
20. Turan, M.S.; Barker, E.; Kelsey, J.; McKay, K.A.; Baish, M.L.; Boyle, M. Recommendation for the entropy sources used for random bit generation. National Institute of Standards and Technology (NIST) Special Publication 800-90B, 2018. doi:10.6028/NIST.SP.800-90B.
21. Mexis, N.; Arul, T.; Anagnostopoulos, N.A.; Frank, F.; Böttger, S.; Hartmann, M.; Hermann, S.; Kavun, E.B.; Katzenbeisser, S. Spatial correlation in weak physical unclonable functions: A comprehensive overview. 2023 26th Euromicro Conference on Digital System Design (DSD); IEEE: Golem, Albania, 2023; pp. 70–78. doi:10.1109/DSD60849.2023.00020.
22. Wang, S.; Song, L.; Chen, W.; Wang, G.; Hao, E.; Li, C.; Hu, Y.; Pan, Y.; Nathan, A.; Hu, G.; Gao, S. Memristor-based intelligent human-like neural computing. *Advanced Electronic Materials* **2023**, *9*, 2200877. doi:10.1002/aelm.202200877.
23. Qin, Y.F.; Bao, H.; Wang, F.; Chen, J.; Li, Y.; Miao, X.S. Recent progress on memristive convolutional neural networks for edge intelligence. *Advanced Intelligent Systems* **2020**, *2*, 2000114. doi:10.1002/aisy.202000114.
24. Milo, V.; Malavena, G.; Monzio Compagnoni, C.; Ielmini, D. Memristive and CMOS devices for neuromorphic computing. *Materials* **2020**, *13*. doi:10.3390/ma13010166.
25. Camuñas-Mesa, L.A.; Linares-Barranco, B.; Serrano-Gotarredona, T. Neuromorphic spiking neural networks and their memristor-CMOS hardware implementations. *Materials* **2019**, *12*. doi:10.3390/ma12172745.
26. Cao, Z.; Sun, B.; Zhou, G.; Mao, S.S.; Zhu, S.H.; Zhang, J.; Ke, C.; Zhao, Y.; Shao, J. Memristor-based neural networks: A bridge from device to artificial intelligence. *Nanoscale Horizons* **2023**. doi:10.1039/D2NH00536K.