Using Memristor Arrays as Physical Unclonable Functions

Florian Frank$^{1[0000-0001-6914-2152]}$, Tolga Arul$^{1,2[0000-0002-2078-3976]}$, Nikolaos Athanasios Anagnostopoulos$^{1,2[0000-0003-0243-8594]}$, and Stefan Katzenbeisser[1]

[1] University of Passau, Innstraße 43, 94032 Passau, Germany {florian.frank, tolga.arul,nikolaos.anagnostopoulos,stefan.katzenbeisser}@uni-passau.de
[2] Technical University of Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Germany {na45tisu,arul}@rbg.informatik.tu-darmstadt.de

Abstract. In this work, we introduce two new types of Physical Unclonable Functions (PUFs) based on memristor arrays. Both PUFs use the output behavior of memristor cells when an excitation signal is applied to their input. First, the cells are identified by decomposing the signal response into different frequencies using the discrete Fourier transformation and evaluating the absolute sum of errors. This approach provides a maximum accuracy of 96% and F1-score of 73%. In order to improve performance, a convolutional neural network is employed to learn the shapes of the output hysteresis loop. To this end, a conversion algorithm that transforms the outputs to matrices is used. The proposed neural network achieves a maximum accuracy of 97% and F1-score of 97%, allowing for the successful utilisation of the examined PUF in practical security applications. As a use case for the proposed PUFs, we introduce a novel neural network-based authentication protocol that can be used to authenticate smart devices to a central IoT hub, e.g., in a smart home.

Keywords: Physical Unclonable Function (PUF) · memristors · machine learning · hardware security · Resistive Random Access Memory (ReRAM) · Convolutional Neural Network (CNN) · neural networks

1 Introduction

The use of microcontrollers influences all areas of life and applications, such as consumer electronics, sensors, and vehicles. These systems are getting smaller and more powerful. At the same time they are often very constrained in their power consumption. Storage technologies employed in these systems will face certain development limits soon due to their integration density and power consumption. One promising technology that could overcome these limitations is **ReRAM** (**Re**sistive **R**andom **A**ccess **M**emory), a non-volatile memory technology, which combines fast switching, low energy consumption, and small cell sizes, without decreasing performance [4]. ReRAMs are based on memristors (**mem**ory and re**sistor**), which are passive circuit elements that change their resistance with the amount of charge floating through them, in relation to their previous resistance value, which is otherwise maintained.

Microcontrollers are often deployed in security-critical areas, such as in-vehicle networks, which makes secure communication between the devices mandatory. Thus, one major security requirement is to establish a secure communication channel between the different system components, which requires device authentication. Many approaches proposed in the literature face the problem that they require the storage of a secret key in the device memory, causing a vulnerability if the attacker has physical access to the device.

One way to solve these problems is to use **PUF**s (**P**hysical **U**nclonable **F**unctions). PUFs generate a digital "fingerprint" of a device, which can be used for authentication and identification. These fingerprints are based on marginal differences in the hardware, which occur during the manufacturing process [9]. The advantage of this method is that the keys do not need to be permanently stored on the device, but can be reproduced on the fly out of certain unique hardware properties right before they are used, which makes them less vulnerable against physical attacks. Different types of PUFs have been proposed: they can be constructed from optical systems [12], ring oscillators [22], or conventional memory modules [8, 18]. The security of PUFs is based on their ability to provide a (usually, binary) pattern that is unique for each device, thus serving as a device identifier [18]. With the continuous adoption of ReRAMs in embedded devices, replacing DRAM modules [24], memristor-based PUFs are becoming more and more appealing as lightweight security primitives. For this reason, in this work, we will examine a PUF implemented on a novel non-volatile memory, namely, a memristor array. Further, the applicability of the novel PUF is demonstrated in the context of a new authentication protocol for the **IoT** (**I**nternet **o**f **T**hings).

1.1 Contributions

The main contributions of this work concern the construction of a novel PUF based on an array of memristor cells, as well as its characterisation, and evaluation, based either on frequency analysis or on machine learning.

More specifically, the PUF is first characterised using a technique based on the frequency composition of the output wave of each memristor cell, when applying a sine wave. We show that, based on the frequency composition, each cell can be uniquely identified by a simple classification of the quantised frequency distribution. To improve the classification performance even further, a second classification method based on **CNNs** (**C**onvolutional **N**eural **N**etworks) is introduced. There, a sine wave is applied to the memristors, causing a continuous change between their high and low resistive states, which results in a so-called "pinched hysteresis loop". We show that this hysteresis is a distinctive feature of memristors, where the shape of the loops differs from cell to cell due to manufacturing variations. CNNs are used to identify the hysteresis loops of such cells, resulting into an accuracy and F1-score of up to 97% and 97%, respectively. This is a significant improvement compared to the analysis by frequency composition that achieves an accuracy of 96% but an F1-score of only 73%.

Furthermore, we demonstrate how this PUF can be used in a smart home to authenticate smart devices like a smart refrigerator, or smart light bulbs, to an IoT hub using a novel authentication protocol tailored for this type of PUF. In general, the presented PUF construction requires access to one or only a few memristor cells to generate a unique pattern that can be used for authentication and identification, and thus is the first of its kind.

1.2 Related Work

The potential of using memristors as PUFs has been explored in several works.

Rose et al. [17] introduced a memristor-based PUF that utilises the differences in the time required for the memristors to transit from the high-resistance to the low-resistance state. The time required for this transition is measured for each memristor and compared to a selected threshold value. If the actual transition time of a memristor is below the threshold, the result is a logical 0, otherwise a logical 1. The set of zeros and ones returned from a particular set of cells constitutes the response of the PUF. In this work, each cell produces only a one-bit response, thus requiring access to many cells to generate a secure key. On the contrary, our PUF requires access to one or only a few memristor cells to generate a unique pattern that can be used for authentication and identification.

Gao and Ranasinghe [7] constructed a PUF, that uses memristors that are arranged in an array-like structure. Each cell in this array consists of two memristors, connected in series. When applying a voltage that is two times the reset voltage, one of the two memristors reaches the off state, i.e., the low-resistance state, first, which causes the second memristor to stop changing its resistance. Afterwards, the memristors are read with a small voltage that does not disturb the resistance of the device. Depending on which memristor stays in the high resistive and which one changes into the low resistive state, a PUF response of a logical 0 or a logical 1 is obtained. The disadvantage of this method is that each memristor pair produces only a one-bit response, which requires a large amount of memristors to produce a secure key, like the work of Rose et al. [17].

Uddin et al. [20] introduced the memristive crossbar PUF (XbarPUF), a PUF based on memristors that uses the switching delays of multiple memristors as the PUF characteristic. An additional PUF based on the resistance differences of two memristors when being in a low or high resistive state, was proposed by Chen et al. [3]. These PUFs require a more complex setup than our implementation, as our PUF requires only a simple measurement generated by connecting a function generator and an oscilloscope to a memristor cell.

Finally, some works have proposed the use of neural networks in the context of PUFs: Yue et al. [23] described an authentication scheme using a deep neural network to extract the unique features from the raw power-up values of DRAM cells, which are then used for authentication. Yilmaz et al. [21] also proposed a PUF-based authentication protocol using neural networks. The delay difference of the neural computation itself was proposed as a PUF characteristic by Nozaki et al. [15], while Najafi et al. [14] proposed a latency-based DRAM PUF that uses neural networks for device identification without the need for error correction.

2 Background

In this section, we provide a brief technical introduction to the functionality of memristors and the properties of Physical Unclonable Functions.

2.1 Self-Directed Channel Memristors

A memristor is a passive circuit element, whose theoretical existence was conceived by Leon Chua as early as 1972, but which was only manufactured around 2008 [19]. A memristor can be described as a resistor with memory. Its resistance at the current state always depends on its resistance at previous states, stored within the device [6]. A memristor cell changes its resistance depending on the amount of charge flowing through it. This behavior is usually shown by applying a sine wave to the memristor and visualizing the output in a Lissajous curve, as demonstrated in Figure 1. Some types of memristors require a forming process, which initializes the chemical and physical structure within the memristor and influences its behavior over its whole lifespan. In our case, a sine wave with an amplitude up to 3 V is applied to the memristor. For our experiments, we are using two memristor arrays of the brand Knowm, each consisting of 16 memristor cells [11]. Further information is given in Appendix 1.

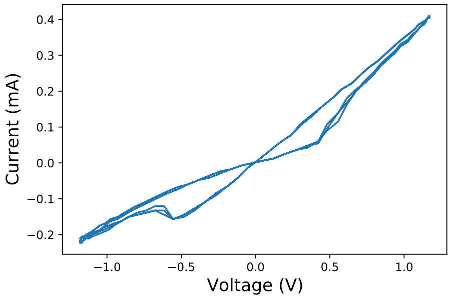

Fig. 1. Lissajous curve of a single memristor cell of the brand Knowm when applying a 100 Hz sine wave with an amplitude of 1.2 V.

2.2 Physical Unclonable Functions

Physical unclonable functions use the hardware properties of a device to produce a unique fingerprint. A PUF accepts a challenge c and returns a corresponding response r, which together form a **C**hallenge-**R**esponse **P**air (**CRP**). For an optimal PUF, the response can only be formed by a specific device, as it originates from physical properties only found in its hardware. For this reason, an ideal PUF is hard to clone and produces unique responses for any given challenge [9].

In addition, a distinction is made in the literature between strong and weak PUFs. Weak PUFs only exhibit one or at most a few CRPs, whereas strong PUFs have a (much) higher number of CRPs available. CRPs can be used for identification and authentication as well as for secure key generation. The use of PUFs has the advantage that no keys have to be permanently stored in physical memory, which could lead to security vulnerabilities when the attacker has physical access to the device [5]. For authenticating an individual device at a later stage, sets of CRPs are gathered during an enrollment phase, right after production. Moreover, commercially available security solutions using PUFs already exist, e.g., PUF-based RFIDs [10] and inbuilt PUFs in Xilinx FPGAs [13].

For the PUFs presented in this work, the challenge consists of an identifier of the memristor cell within the array, as well as the amplitude and the frequency of the input sine wave. The PUF response consists of the hysteresis loop produced by the memristor cell under the input sine wave used in the challenge.

To assess the quality of the examined PUFs, the most important properties are **Uniqueness,** which measures the independence of responses originating from multiple PUF instances for the same challenge c, and **Reliability,** which describes the stability of PUF responses, for a given challenge c, under repeated PUF measurements. Typically, these properties are measured by metrics based on the Hamming distance or the Jaccard index. These classical metrics are not applicable to our PUF implementations, because these PUFs are evaluated using classification techniques that are rather fuzzy.

Since we pursue an approach that employs machine learning to assign responses to corresponding challenges, we use the accuracy and F1-score metrics to assess the performance of the classification and ultimately rate the quality of the resulting PUF. Here, the metric of accuracy represents the number of correctly classified memristor cells over the total number of PUF instances. This metric provides an indication of how well the classification is working, but is insufficient because our data have an uneven class distribution.

For this reason, we additionally use the F1-score, which considers further aspects of the data set such as its recall, precision and false positives. Both metrics are examined in more detail in Section 3.2.2.

3 Memristance-Based PUFs

3.1 Measurement Circuit Design

We investigate the effects of the frequency and amplitude of the input sine signal on the memristive behavior, since these quantities are used to form the challenge for the memristance-based PUFs we propose in this work. To examine the electrical characteristics of the memristor cells, a Keysight 33500B function generator and a Keysight MSOX3104T oscilloscope are used. The function generator is connected to the input of the memristor cell and can apply a sinus wave to it. The output is connected to a resistor to limit the current. The first channel of the oscilloscope captures the output of the signal generator, while the second channel measures the voltage drop across the shunt resistor to calculate the current resistance of the memristor. Finally, the measurement devices and the memristor are connected to a common ground. Our experiments have been performed using the parameters given in Table 1, because these provide the best evaluation results to uniquely identify single memristors. More information about the measurement circuitry is provided in Appendix 2.

Table 1. Parameter values used for testing and capturing the behavior of the memristor cells. Combinations of these values are used for evaluating the PUF.

Parameters	**Values**
Frequency	{100 Hz, 500 Hz, 1 kHz, 10 kHz}
Amplitude	{0.8 V, 1.0 V, 1.2 V, 1.5 V}

3.2 Classification of Memristor Cells Based on their Frequency Distribution

The memristor PUFs considered in this work are based on the characteristic memristance of each device, which is caused by differences in the movement of Ag^+ ions into and from the active layer. Since the measured voltage drop is inversely proportional to the resistance of the memristor and the applied input voltage, we can use memristance to distinguish individual memristor cells.

3.2.1 PUF construction After capturing the measurement data, we observed that each memristor cell produces a hysteresis loop with a unique shape, which we use to identify each memristor cell of a memristor array. Figure 2 shows the unique shape of 16 memristors, each in a different color. The figure shows the input voltage of the memristor cell, V_{in}, on the x-axis, and the voltage drop occurring after the memristor, V_{out}, on the y-axis.

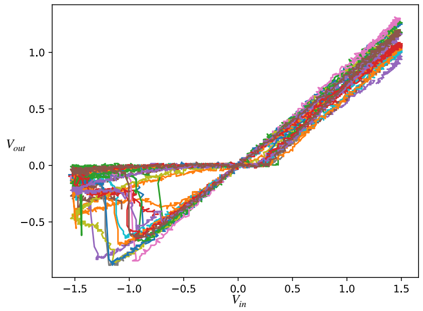

Fig. 2. Hysteresis loops of 16 different memristor cells (each cell is represented by one color). All measurements are performed using an input sine wave with a frequency of $F_{in} = 100\,\text{Hz}$ and an amplitude of $A_{in} = 1.2\,\text{V}$.

First, an enrollment is executed where 200 sine waves of V_{in} and V_{out} are captured. Afterwards each curve is sampled at 100 points. There, the sampling rate is high enough to distinguish the cells based on their unique properties. With $F_{in} = 100\,\text{Hz}$, only one measurement per 100 µs must be captured, which could also be done on very resource-constraint systems. Using 200 of those samples during the enrollment allows us to capture differences from one cycle to the other. This allows us to make our classification method more robust when classifying further measurements with small deviations.

A Discrete Fourier Transformation is applied to the two sets of waves V_{in} and V_{out} separately, resulting in a frequency spectrum from 0 Hz to f^n, sampled in steps of size s:

$$FS_{in}(s, f^n) = DFT(s, f^n, \{V_{in}^0, \ldots, V_{in}^{199}\}) \, ,$$

$$FS_{out}(s, f^n) = DFT(s, f^n, \{V_{out}^0, \ldots, V_{out}^{199}\}) \, .$$

There, the 200 measurements are treated as continuous waves resulting in two frequency spectra: $FS_{in} := \{f^0_{in}, ..., f^n_{in}\}$ and $FS_{out} := \{f^0_{out}, ..., f^n_{out}\}$ containing the samples of each frequency f from $f^0 = 0\,\mathrm{Hz}$ to f^n, where f^n is the maximum frequency. In our case, $f^n = F_{in} * 10$, where F_{in} is the applied input frequency, because higher-level characteristics of the loop, like the dent of the hysteresis, only occur at frequencies ranging from $F_{in} * 4$ to about $F_{in} * 10$. The step size s is defined as $f^n/1000$. The frequency spectrum is subdivided into 1000 steps, which results into a good trade-off between having a high enough resolution and not generating too many data. The intervals are further optimized by subdividing them into chunks, as described later.

In the next pre-processing step, noise and the dominant frequencies caused by V_{in} are removed from V_{out} by subtracting each of the 1000 samples in the input frequency spectrum from the output spectrum: $FS_{res} = FS_{out} - FS_{in}$. The resulting decomposition, FS_{res}, can then be used as an identification feature for a specific memristor cell.

Subsequently, the most characteristic frequency ranges are extracted from FS_{res}, which enables the most accurate classification. Figure 3 depicts the influence of different frequencies on the shape of the hysteresis loop, when applying a 100 Hz input sine wave to a memristor. On the left side, a captured hysteresis loop without any post-processing is shown. On the right side, all frequencies $f > 400\,\mathrm{Hz}$ are removed from the frequency spectrum of FS_{in} as well as FS_{out}, resulting in a smoothed hysteresis loop. We note that by removing frequencies higher than 400 Hz, the small dent that can be seen on the left side of the loop, which is a very distinctive characteristic for each individual cell, is removed.

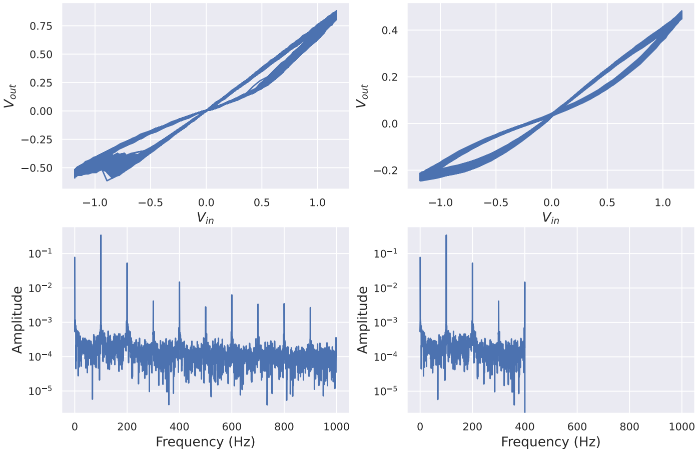

Fig. 3. Left: hysteresis loop collected from the memristor array (top), and its corresponding frequency spectrum (bottom). Right: all frequencies $f > 400\,\mathrm{Hz}$ have been removed from the frequency spectrum (bottom); the inverse discrete Fourier transformation results in a smoothed hysteresis loop (top).

Thus, we distinguish the hysteresis loops based on the samples of both the lower and the higher, i.e., of only the outermost, frequency regions. The lower frequencies are responsible for the basic hysteresis shape, whereas higher frequencies account for smaller edges and structures, like the characteristic dent shown in Figure 3. The combination of these two frequency regions results in a unique characteristic for each memristor cell.

For that reason, the frequency spectrum FS_{res} is filtered, so that f_{res}^{199} to f_{res}^{398} are removed from the spectrum. This frequency spectrum is chosen because the basic shape of the hysteresis curves is generated by the lower frequencies of f_{res}^{0} to f_{res}^{199}, while more specific forms, like the characteristic dent, occur at frequencies higher than f_{res}^{399}, e.g., when applying 100 Hz, the basic shape is generated by frequencies up to 200 Hz, while the characteristic dent of the loop is generated by frequencies higher than 400 Hz.

The remaining spectrum, FS_{filter}, is then subdivided into c chunks. The average value of each chunk of each memristor on the enrollment measurements is then calculated for all measurements, and used as a reference for further classification. The optimization of the width of c is described in Section 3.2.2.

During verification, a PUF measurement is performed and pre-processed in a similar way as during enrollment. Finally, the absolute error, defined by:

$$e = \sum_{i=1}^{n} |y_i - \hat{y_i}| \,,$$

is calculated between the chunks of the new measurement and the chunks calculated during the enrollment. In this formula, y_i corresponds to a chunk i in FS_{filter} calculated during the enrollment phase, and $\hat{y_i}$ to a chunk i of the new measurement m'.

Each new measurement is identified as corresponding to the cell for which the minimum absolute error occurs. In the heatmap in Figure 4, the absolute error between each measurement during the enrollment and later measurements is shown. On the right side, the minimum values are visualized in yellow color. It can be seen that, for most cells, the absolute error results in the lowest distance when comparing the frequency spectra of measurements of the same cell, which leads to a correct assignment most of the time.

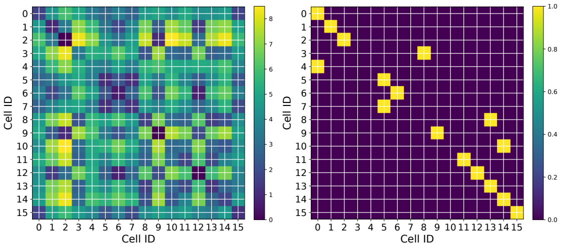

Fig. 4. The left image shows the absolute error values between measurements from all the cells. On the right side, the minimum value of each row is shown. Here, 11 out of 15 working cells (cell 3 is damaged) are stable, meaning that the relevant measurements can be assigned to the correct cell, leading to only 4 false positives (the false positive for cell 3 is not counted) and 4 false negatives. For the use of the PUF in practical applications, only cells producing stable responses are used.

3.2.2 Evaluation In this work, we utilise the concept of *accuracy* and the *F1-score* to measure the PUF properties of uniqueness and reliability which are more suitable for this classification problem in comparison to the traditional PUF metrics of the Hamming distance and Jaccard index.

Note that by testing how accurately responses originating from a particular memristor cell can be assigned to it (rate of true positives – TP) and how accurately responses originating from different PUFs can be identified as not originating from that particular PUF (rate of true negatives – TN), we can easily get a single metric that reflects both reliability and uniqueness. A high intra-class accuracy (TP) indicates that measurements originating from each PUF instance can correctly be attributed to it (a high level of reliability), while a high inter-class accuracy (TN) indicates that measurements originating from different PUF instances can be correctly attributed to the correct memristor cell. A high degree of accurate classification, however, is only possible if the relevant measurements are highly distinguishable, therefore reflecting also a high level of uniqueness. Thus, by evaluating the examined PUF instances with the metrics of accuracy and the F1-score, we are able to provide a simple, yet practical and efficient, way in which the overall quality of these PUFs can be estimated.

Based on the classification and assignment of responses to memristor cells, we can calculate the relevant rates of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), to obtain the accuracy as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \, . \tag{1}$$

In addition, we use the F1-score, which combines the precision and recall of the data set into a single metric, because we consider it a more appropriate metric due to the number of false positives we observed during our experiments. The F1-score is calculated using the equation:

$$F1\text{-}score = \frac{TP}{TP + 0.5 * (FN + FP)} \, . \tag{2}$$

These metrics allow us to measure how often a measurement is assigned to the correct cell in proportion to all assignments, and thus also describe how reliable our identification scheme is.

As expected, the cells on the diagonal of the heatmap of Figure 4, comparing measurements of the same memristor cell, mostly show the lowest error values. This means that measurements corresponding to the same cell are indeed very similar. First, the number of stable cells is calculated. Such cells can be identified based on the difference between the chunks of the histogram. We note that some cells of the analyzed memristor chips were damaged and thus provide unstable responses. These cells are detected and removed from the measurement sets. The PUF is evaluated using the parameters described in Section 3.1. The classification performance, including the F1-score and the accuracy metrics, is shown in Table 2.

At most 11 out of 15 cells can be identified. One of the 16 cells of each memristor array is identified as damaged and is consequently removed from the data set. As Figure 4 illustrates, in the best case, this method leads to only 4 false positives and 4 false negatives, resulting in the following values:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{all - (FP + FN)}{all} = \frac{(15 * 15) - 8}{(15 * 15)} \approx 96\%\,,$$

$$F1\text{-}score = \frac{11}{11 + 0.5 * 8} \approx 73\%\,.$$

However, Chip 1 exhibits a higher degree of instability, and therefore provides worse results. Since the expansion of the hysteresis loop becomes smaller by increasing the input signal frequency and lowering the voltage amplitude, generally also the differences among the characteristics of the cells become smaller, leading to worse results. For that reason, measurements with a frequency of 100 Hz and an amplitude of 1.2 V show the best results. The frequency distribution is subdivided into n different chunks. Selecting the amount of chunks is part of the hyper-parameter optimization. Having more chunks preserves more detail in the frequency distribution, whereas smaller chunk sizes allow the reduction of noise from the frequency spectrum. The best chunk size for each combination of parameters, which most often is 92, can be seen in Table 2.

Table 2. Number of correctly classified cells, among all undamaged cells on different amplitudes with a frequency of 100 Hz.

Chip ID	**Frequency in Hz**	**Amplitude in V**	**# Chunks**	**# Correctly Classified Cells**	**# Undamaged Cells**	**Acc**	$\mathbf{F_1}$
1	100	0.8	92	6	15	0.92	0.4
1	100	1.2	92	7	15	0.93	0.47
1	100	1.0	200	7	15	0.93	0.47
2	100	0.8	92	10	15	0.96	0.67
2	100	1.0	94	9	15	0.95	0.6
2	100	1.2	92	11	15	0.96	0.73

3.3 Classification of Memristance-Based PUFs Using Convolutional Neural Networks

In order to obtain a method that achieves higher values for the F1-score, we propose a classification method based on convolutional neural networks. These are able to learn the discrete shape of the hysteresis curves of the memristors, after they have first been transformed into pixel images. The huge advantage of this type of neural network is that it can learn local spatial coherence. The transformation of the curves into pixel images is a prepossessing method that allows for reducing noise through quantization and mitigates the problem of overfitting. As shown in Section 3.3.2, the learned patterns of the CNN can be visualized, which allows tracing which shapes are learned, which is a huge advantage in comparison with other types of neural networks.

3.3.1 PUF construction CNNs were developed specifically for the domain of computer vision and are very suitable for image processing tasks, such as image classification. The complexity of shapes that can be learned by CNNs increases with the number of layers. For example, using only one layer, only simple edges can be learned. Adding a few more layers allows to recognize objects within a picture, and by adding additional layers complex and more detailed structures within pictures can be learned. The size of the local learnable patterns is specified by the size of a kernel filter. For instance, a filter with size of 5×5 pixels iterates over the image and can only learn local shapes of that size. However, this has the advantage that fewer weights are needed in comparison to densely connected neural networks, due to weight sharing, the calculation on local patterns, and the usage of max-pooling layers . In the next layer, a new 5×5 kernel filter can learn more complex shapes by operating on the output of the previous layer. Additional max pooling layers are required to reduce the size of the images after each convolution layer. Here, the maximum value of the kernel filter is selected so as to reduce the number of weights of the next layer [1]. Finally, a **ReLU** (**Re**ctified **L**inear **U**nit) activation function as well as a softmax layer are attached to classify single memristors within the memristor array. In the first step, the data captured from the memristor cells are transformed into pixel images. There, w describes the number of pixels in horizontal direction, and h the height of the image. Selecting w and h is part of the hyperparameter optimization and is described in the subsequent section.

Additionally, the minimum and maximum values of the input voltage, $min(V_{in})$ and $max(V_{in})$ respectively, are determined. Then, the range $r_x := \{min(V_{in}), max(V_{in})\}$ is subdivided into w bins. The same is done for the output channel V_{out}. Here, the range $r_y := \{min(V_{out}), max(V_{out})\}$ is subdivided into h bins. Subsequently, all the V_{in} values of one measurement are assigned to the corresponding w bins, based on their position in the range r_x. The same is done for V_{out} values and the corresponding h bins, based on r_y. The combination of w vertical and h horizontal bins results in a matrix, which can be learned by the CNN. If multiple values are assigned to one chunk, the number of values in each chunk is stored, which increases the performance of the neural network. Such a transformation is visualized in Figure 5: The right side shows the hysteresis loop that is shown on the left side, having been transformed into a matrix, forming a pixel image.

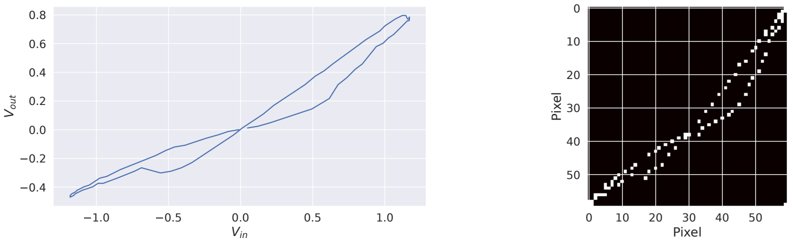

Fig. 5. Left: Hysteresis loop of a memristor. Right: Visualization of the transformation of the hysteresis loop into a pixel image.

In this way, we transform the PUF measurement data into 60×60 pixel images, and further optimize w and h. This allows us to maintain a high degree of detail while reducing noise and limiting the amount of data, so that the analysis can be handled in a short amount of time. In the next step, all converted pixel images are combined into a single tensor. For each measurement, a label specifying the corresponding memristor cell, by using a cell ID implemented as a one-hot encoded vector, is created.

Our CNN consists of a 2D convolutional layer as the input layer, using a 3×3 kernel filter. This layer is followed by a MaxPooling layer that operates on the output of the previous layer by using a 2×2 kernel filter. In total, we stack three convolutional and MaxPooling layers. In the end, the output is flattened and fed into a densely connected layer with 64 neurons and a softmax classifier to learn the different classes, each corresponding to one memristor in the memristor array. An outstanding advantage of convolutional neural networks is that they can visualize the learned features. In Figure 6, we can see that the neural network distinguishes cells specifically by considering the area of the highest and lowest voltage, which correspond to the most characteristic patterns of the hysteresis loop used for the identification of the cells.

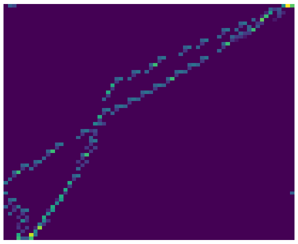

Fig. 6. Visualization of the fourth convolutional layer of the neural network. This heatmap shows the significance of the different sections of the hysteresis curve used by the network for distinguishing memristor cells.

3.3.2 Evaluation First, multiple CNNs are trained to determine the best ratio of pixel width and height to achieve the best accuracy, F1-score and the lowest loss. It is particularly important to find the best width and height values so that a high level of detail can be preserved, and noise from the measurement can be reduced. The resulting pixel image is processed in layer four of the CNN as depicted in Figure 6.

Here, pixels with a brighter hue are of special interest for the CNN and have a greater influence on distinguishing the different samples. Afterwards, the pixel width and height, the amount of layers, the batch size, the number of training epochs, as well as the ratio of the partition of the data into training, validation, and test data are optimized in order to reach the highest accuracy and F1 values. According to our evaluation, a convolutional neural network with five layers provides the best performance over all frequency ranges and amplitudes.

For each amplitude and frequency, the hysteresis loops of each memristor cell are trained over 20 epochs with a batch size of 5. Table 3 shows the results when testing the neural network with hysteresis loops not seen during training. The highest accuracy of 97% and the highest F1-score of also 97% are achieved when the data set of memristor Chip 2 is trained on data where the input signal has a frequency of 100 Hz and an amplitude of 1.0 V. Under these conditions, 97 out of 100 samples were correctly classified, demonstrating the high reliability of the memristor cell classification scheme. We have additionally tried to classify the cells with a densely connected neural network operating on the voltage arrays V_{in} and V_{out}. Thereby, we could achieve an accuracy and F1-score of 94%, in the best case, which may not be sufficient for an authentication application.

As expected, the accuracy of the classification is decreasing with higher frequencies, since, in this case, the hysteresis loop has a lower expansion, and noise has a higher impact on the measurements. We observe that the memristor Chip 2 performs better than Chip 1, which can be attributed to the presence of more unstable cells in Chip 1. Again, as expected, Chip 2 delivers the worst values at the highest frequency, caused by a decreased expansion of the hysteresis loop. However, Chip 1 deviates from this behavior since the best performance here is achieved using an input signal with a frequency of 1 kHz and an amplitude of 1 V. In the future, we plan to investigate whether the combination of multiple memristors to identify a particular array, and hyper-parameter optimization, can lead into an increased accuracy and F1-score.

Table 3. Accuracy and F1-scores for all the amplitude and frequency combinations used to train the convolutional neural network. Each amplitude and frequency combination corresponds to a particular input wave.

Chip ID	Frequency in Hz	Amplitude in V	x-Dim in px	y-Dim in px	# Epochs	Acc	F1
1	100	0.8	80	80	20	0.91	0.91
1	100	1.0	150	50	20	0.76	0.76
1	100	1.2	80	80	20	0.78	0.78
1	500	0.8	150	50	20	0.78	0.78
1	500	1.0	80	80	20	0.80	0.80
1	500	1.2	80	80	20	0.80	0.80
1	1000	0.8	80	80	20	0.78	0.79
1	1000	1.0	80	80	20	0.86	0.86
1	1000	1.2	110	50	20	0.88	0.87
2	100	0.8	60	60	20	0.92	0.92
2	100	1.0	120	50	20	0.97	0.97
2	100	1.2	60	60	20	0.94	0.94
2	500	0.8	60	60	20	0.86	0.86
2	500	1.0	60	60	20	0.85	0.86
2	500	1.2	60	60	20	0.92	0.92
2	1000	0.8	60	60	20	0.84	0.84
2	1000	1.0	60	60	20	0.91	0.91
2	1000	1.2	60	60	20	0.79	0.79

4 Applications of Memristance-Based PUFs

The low resource requirements of memristor PUFs, such as their component cost, processing overhead, and power consumption, allow them to be used in a variety of different applications. In particular, the proposed PUFs are suitable for securing and authenticating end devices in a smart home (see Figure 7), where low resource requirements are critical for the successful adoption of solutions.

The proposed PUF-based protocol is lightweight, as for the CNN model only the relevant node weights need to be stored. Its design is kept as generic as possible to increase its compatibility with other IoT technologies and to be able to later adopt this protocol also for other scenarios. Additionally, the protocol is suitable for resource-constrained devices because the major computational effort of training the model is done by the manufacturer. There, the training of multiple devices can be done in parallel. Thus, the biggest overhead occurs at production time. During operation, only the IoT hub needs to evaluate the measurements provided by each device. This is done with only one forward propagation through the neural network that does not require significant resources and can be executed quickly, causing only a small delay to the authentication process. Even if the frequency-based method is used in this protocol, the resource-constrained devices only need to provide the measurements to the IoT hub, which is responsible for the calculation of the Fourier transformation and the classification.

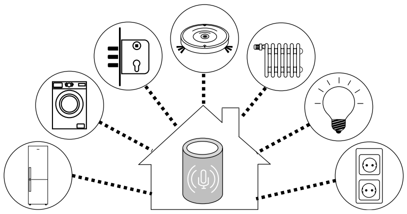

Fig. 7. Typical components of a smart home.

4.1 Authentication Protocol

For the above-mentioned scenario, we propose an authentication protocol that consists of two phases, as depicted in Figure 8. In the **enrollment phase,** measurements $M_{D_{ID}} := \{M_{C_0}, \ldots, M_{C_n}\}$, each corresponding to a memristor cell from C_0 to C_n of a smart device with an identifier D_{ID}, are captured and transmitted to the IoT Hub over a secure channel, e.g. by establishing a direct connection between the IoT Hub and the device in a controlled environment without any other network connection. Each cell is measured with multiple frequencies $F := \{f_0, ..., f_n\}$ and amplitudes $A := \{a_0, ..., a_n\}$. The IoT Hub stores the model $Model_{D_{ID}}$ of the smart device learned by the manufacturer along with an identifier D_{ID}. In the highly unlikely case that the CNN model fails to be produced for a particular D_{ID} after a few attempts, this D_{ID} shall not be used. Also, a public key is transmitted to the smart device, which is used in the next steps of the protocol. There, lightweight algorithms using elliptic curve cryptography are used.

In the **authentication phase**, a challenge-response protocol is executed. First, the smart device sends a challenge with its identifier D_{ID} to start an authentication request. Here, || describes the concatenation of the message, and the first italicised segment, e.g., *AuthRequest*, is an identifier, allowing the IoT Hub and the device involved to identify and parse the relevant messages correctly. The server responds with a challenge containing a device ID D'_{ID}, a nonce N, a cell ID C_{ID}, an amplitude a, and a frequency f. N is used to prevent replay attacks and can be implemented as a continuous counter or a random number. The IoT device first checks if the requested device ID, D'_{ID}, is equal to its own, and then measures the cell C_{ID} by applying the frequency f and amplitude a to it, resulting in a measurement M_{CID}. Afterwards, a message consisting of the D_{ID}, the previously sent nonce N, and the measurement M_{CID} is encrypted using the previously shared *publicKey*, and sent to the IoT Hub. Only the IoT Hub can decrypt the message using its *privateKey*. The server checks if N is fresh, and chooses the right model, $Model_{D_{ID}}$, based on D_{ID}. If the CNN can classify all measurements correctly using $Model_{D_{ID}}$, a, and f, the IoT device gets authenticated, otherwise it gets rejected. After a number of unsuccessful authentication requests for the same D_{ID}, the use of that D_{ID} may be disabled.

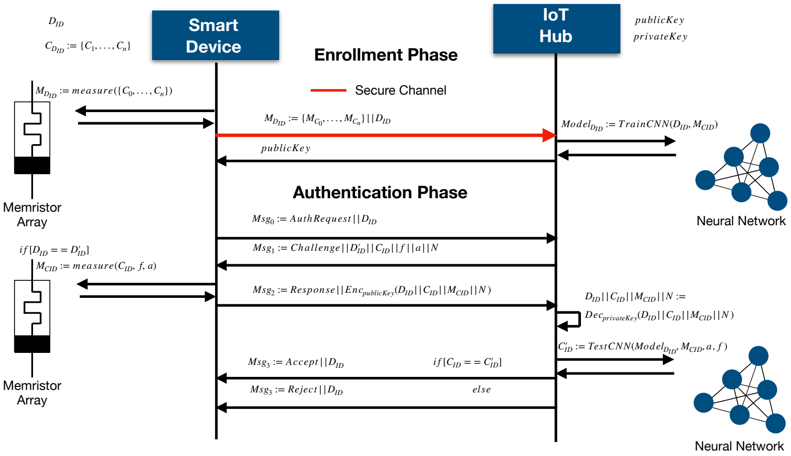
D_{ID}
$C_{D_{ID}} := \{C_1, \ldots, C_n\}$
Smart Device
IoT Hub
$publicKey$
$privateKey$
Enrollment Phase
Secure Channel
$M_{D_{ID}} := measure(\{C_0, \ldots, C_n\})$
$M_{D_{ID}} := \{M_{C_0}, \ldots, M_{C_n}\}||D_{ID}$
$Model_{D_{ID}} := TrainCNN(D_{ID}, M_{CID})$
$publicKey$
Memristor Array
Neural Network
Authentication Phase
$Msg_0 := AuthRequest||D_{ID}$
$if[D_{ID} == D'_{ID}]$
$M_{CID} := measure(C_{ID}, f, a)$
$Msg_1 := Challenge||D'_{ID}||C_{ID}||f||a||N$
$Msg_2 := Response||Enc_{publicKey}(D_{ID}||C_{ID}||M_{CID}||N)$
$D_{ID}||C_{ID}||M_{CID}||N :=$
$Dec_{privateKey}(D_{ID}||C_{ID}||M_{CID}||N)$
$C'_{ID} := TestCNN(Model_{D_{ID}}, M_{CID}, a, f)$
$Msg_3 := Accept||D_{ID}$
$if[C_{ID} == C'_{ID}]$
$Msg_3 := Reject||D_{ID}$
$else$
Memristor Array
Neural Network

Fig. 8. The proposed protocol comprises two phases: a one-time enrollment phase, and an authentication phase that may be executed multiple times, as needed.

4.2 Evaluation of the Proposed Protocol

The advantage of the proposed protocol is that the memristor measurements do not have to be stored on any of the devices. The shared secret only relies on the CNN model that is stored on the side of the central entity, and the ability to reproduce the memristor measurements on the IoT device side.

Our adversary model for the proposed protocol considers a passive attacker who is able to observe the network traffic between the smart device and the IoT hub, and who can capture transmitted messages. Furthermore, we consider the machine learning parameters, such as the CNN architecture, but *not* the relevant node weights, to be public, and thus known by the attacker.

Therefore, an attacker is not able to retrieve the measurements from the response message without knowing the $publicKey$. Also replay attacks can be detected, by checking if a nonce occurred twice on the server's side. An attacker is not able to change N, because the memristor measurements, which are part of the encrypted message, are not known Even if an attacker has physical access to the server, only the CNN model can be retrieved and no measurement data, as this would require reverse engineering the model. In a more realistic scenario, the memristor circuitry may need to be shielded to prevent attacks, for example, by electromagnetic interference, which could potentially disturb the measurements.

In a practical application, a much larger memristor array should be used from which different subsets are selected as a challenge. This would significantly improve the security, increase the challenge space, and thus may form a strong PUF.

5 Conclusion

In this work, we have proposed two methods to generate PUFs based on the characteristic response of memristor cells to alternating voltage, i.e., the hysteresis loop produced. In our first approach, we have analyzed the frequency distribution of the hysteresis loop. By making use of the relevant frequency bands, we could achieve an accuracy of 96%, but an F1-score of only 73%. Subsequently, we have employed neural networks for the classification of PUF responses. Here, we took the approach of identifying individual memristor cells based on the characteristic shape of their hysteresis curve. We were also able to determine which particular parts of the hysteresis curve contain the information most essential for the identification of an individual cell. This knowledge could be used in future work to design another analytical method besides the frequency analysis proposed in this paper. The use of convolutional neural networks could accomplish an accuracy of 97% and an F1-score of 97%. Our investigation shows that the general quality of the proposed PUFs decreases when the frequency of the input signal increases or its amplitude decreases. A more detailed evaluation of memristor-based PUFs will be done in the future. There, also the approaches described in Section 1.2 will be compared to our PUF-based scheme in terms of uniqueness and robustness, to further evaluate our work. In addition, further post-processing techniques and more advanced ML schemes could be used to increase the accuracy and F1-score achieved by the proposed frequency analysis method. Another research direction would be to consider the effects of external factors, such as ambient temperature, as well as the effects of different material compositions and ageing on the quality of the examined PUF.

Acknowledgements. This work has been funded by the DFG, under Projects 440182124 and 439892735 of SPP 2253. The authors would like to thank the anonymous reviewers and the shepherd of this work, Prof. Dr. Özgür Sinanoğlu, for their comments, suggestions, and guidance.

Appendix 1 Self-Directed Channel Memristors

Memristors are passive circuit elements whose behavior can be described by the following simplified equations:

$$v = R(w)\, i, \quad \frac{dw}{dt} = i.$$

Here, the voltage v depends on the current i and the resistance R of the memristor, which in turn depends on the previous state w of the memristor. i can be described as the integral of w over time t, which means that w is essentially the charge that has moved through the memristor [19]. Therefore, a memristor cell changes its resistance depending on the amount of charge flowing through it. This behavior is usually shown by applying a sine wave to the memristor and visualizing the output in a Lissajous curve, as demonstrated in Figure 1.

The memristor cells used in our work are so-called self-directed channel memristors [11]. There, each cell consists of multiple layers. The most important ones are the active layer, and a layer of silver, from which Ag^{+} ions can migrate into the active layer. As visualized in the simplified structure of such a cell in Figure 9, each cell has a top and a bottom electrode, over which voltage can be applied.

By applying a positive potential to the electrode pair, the memristor performs a transition into a low-resistance state; a transition into a high-resistance state is performed when a negative potential is applied. The active layer of the memristor consists of an amorphous chalcogenide using tungsten as a dopant ($W + Ge_2Se_3$). This layer consists of Ge-rich chalcogenide glass, which builds a network connected by Ge-Ge bonds. In this layer, the resistance is controlled by the number of silver ions of the Ag^+ layer that have migrated into this layer (the active layer), and, in consequence, by whether a conductive channel across this layer exists. The amount of ions of silver that migrate into the active layer depends on the potential applied between the top and bottom electrodes.

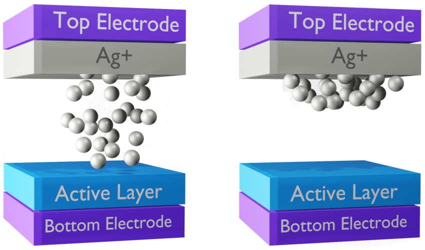

Fig. 9. Simplified structure of a self-directed channel memristor cell. The left image shows the migration of a large number of Ag^+ ions into the active layer, which leads to a highly conductive channel. This is caused by the application of a positive potential to the electrodes. The right image shows the memristor in a high resistive state, after a negative potential has been applied. Here, the Ag^+ ions move from the active layer back to the Ag^+ layer, which leads to low conductance, as a conductive channel is no longer formed across the active layer.

During the initial operation of a memristor, a preliminary step called forming must be executed. Here, the same positive potential as the one used during normal operation, is applied to the top and bottom electrodes. This leads to self-trapped electron pairs around the Ge-Ge bonds, causing, as a reaction, some of the Ge-Ge bonds to break, and Se ions of an adjacent SnSe layer, which is not shown in Figure 9 for reasons of simplicity, to be forced into the active layer. This reduces the energy required to substitute Ag for Ge in a Ge-Ge bond, leading to a conductive channel. The number of Ag^+ ions being forced into the active layer depends on the positive potential applied to the electrodes. Ag^+ ions are removed from the active layer when a negative potential is applied [2].

This ion migration allows each memristor to be used as a memory cell. In order to change to a low resistive state, a voltage V_{SET} must be applied, which means that a positive voltage pulse above a certain threshold must be applied to the memristor. This leads to a migration of Ag^+ ions into the active layer and thus reduces the resistance of the memristor cell. When a negative pulse V_{RESET} that is beyond a certain threshold, is applied, the conductive ions are removed from the active layer and transferred back to the Ag^+ layer, which increases the resistance of the memristor. When a voltage between these two thresholds is applied, the ions stay at their position and the resistance either does not change or does so only to a relatively small extent. In this way, the current resistance of the memory can be measured. The states of high and low resistance encode the logical value of 0 and 1, respectively. When unplugging the power supply, the ions stay at their current position, which makes these cells non-volatile [7].

Appendix 2 Measurement Circuit Design

We designed the measurement circuit in a way such that the experiments are reproducible and measurement data can be captured with a high degree of detail. For that reason, the experiments are performed in an automated test environment and executed by remotely controlling the function generator and oscilloscope. This allows us to capture the data as precisely as possible and thus get consistent and reproducible measurements for all memristor cells. The experimental setup is illustrated in Figure 10.

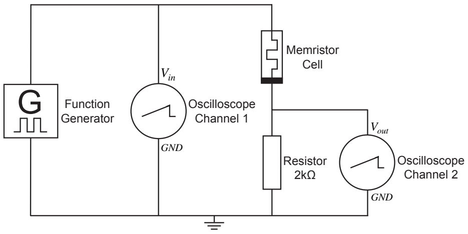

Fig. 10. Circuit used to capture the data using a function generator and an oscilloscope with two channels.

The output connectors of the function generator are connected to the top electrode of the memristor array and to the ground. For measuring different cells in the memristor cell array, this connection is sufficient since the top electrodes of all cells are internally interconnected on the chip carrying the cell array. The first channel of the oscilloscope is connected in parallel to the function generator to capture its output V_{in}. Again, in parallel to the function generator, a resistor and the memristor under test are connected in series. The resistor is used to limit the current floating through the memristors. It is rated with 2 kΩ to limit the current to a maximum of 1 mA when applying up to 2 V. Otherwise, the memristor could suffer damage and remain in a high resistive state permanently [11]. Moreover, the resistor is used to measure the voltage drop caused by the resistance of the memristor. For this purpose, the second channel of the oscilloscope is connected in parallel to the resistor and captures the voltage drop caused by memristor V_{out}. Finally, the ground connectors of the two oscilloscope channels, the ground pin of the function generator, and the resistor are connected to one common ground. The oscilloscope is switched to X/Y mode to visualize the output of the first channel on the x-axis, and the second channel on the y-axis. The constant resistance change of the memristor, due to the sine wave applied by the signal generator, causes a perpetually changing voltage drop at the resistor. Before starting the measurements, a forming operation described in Section 2.1 needs to be performed. During this phase, the 2 kΩ resistor is replaced by a 10 kΩ resistor, to follow the forming process described by the manufacturer of the chip [16]. The 10 kΩ resistor allows us to apply a higher maximum voltage of 3 V to the memristor. Here, the function generator is set to supply a sine wave with 100 Hz and an amplitude of 250 mV. The amplitude is slowly increased up to 3 V, which is the maximum voltage for this type of memristor chip. All subsequent tests are performed with a maximum amplitude of 1.2 V using a 2 kΩ resistor.

A hysteresis loop can be observed at any frequency and an amplitude of 700 mV, for most of the cells. The loop with the greatest expansion on the y-axis can be seen when amplitudes from 1.2 V to 2 V are supplied using a 2 kΩ resistor. Also, the frequency of the sine wave influences the shape of the loop. Lower frequencies lead to more distinctive hysteresis loops. With increasing frequency, the memristors exhibit smaller resistance changes and thus a smaller hysteresis.

Input signals with an amplitude of 1.2 V and frequencies between 100 Hz and 500 Hz almost consistently produce a clear hysteresis loop. When the frequency is increased to 1 kHz, the width in y-direction is getting smaller. By supplying a frequency of 10 kHz, the width of the hysteresis loop gets considerably smaller.

We have also investigated the influence of different amplitudes of the sine signal on the memristive behavior. When using an amplitude of 0.8 V, almost all cells exhibit a hysteresis loop. With decreasing amplitudes, the cells adjust their behavior to that of ordinary resistors. By increasing the voltage, the hysteresis loop expands not only in the x-direction, but also in y-direction. The clearest shape can be seen using amplitudes between 1.2 V and 2 V using a 2 kΩ resistor. The most significant differences among the hysteresis loops of the memristor arrays also arise for these parameter values, which allows us to identify the cells used for our PUF with the highest precision.

References

1. Albawi, S., Mohammed, T.A., Al-Zawi, S.: Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET). pp. 1–6 (2017). https://doi.org/10.1109/ICEngTechnol.2017.8308186
2. Campbell, K.A.: Self-directed channel memristor for high temperature operation. Microelectronics Journal **59**, 10–14 (2017). https://doi.org/10.1016/j.mejo.2016.11.006
3. Chen, A.: Utilizing the variability of resistive random access memory to implement reconfigurable physical unclonable functions. IEEE Electron Device Letters **36**(2), 138–140 (2015). https://doi.org/10.1109/LED.2014.2385870
4. Chen, Y.: ReRAM: History, status, and future. IEEE Transactions on Electron Devices **67**(4), 1420–1433 (2020). https://doi.org/10.1109/TED.2019.2961505
5. Chen, Y., Petti, C.: ReRAM technology evolution for storage class memory application. In: 2016 46th European Solid-State Device Research Conference (ESSDERC). pp. 432–435 (2016). https://doi.org/10.1109/ESSDERC.2016.7599678
6. Chua, L.: Memristor—The missing circuit element. IEEE Transactions on circuit theory **18**(5), 507–519 (1971). https://doi.org/10.1109/TCT.1971.1083337
7. Gao, Y., Ranasinghe, D.C.: R^3 PUF: A highly reliable memristive device based reconfigurable PUF. arXiv preprint (2017), `http://arxiv.org/abs/1702.07491`
8. Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P.: Physical unclonable functions and public-key crypto for FPGA IP protection. In: 2007 International Conference on Field Programmable Logic and Applications. pp. 189–195 (2007). https://doi.org/10.1109/FPL.2007.4380646

9. Herder, C., Yu, M.D., Koushanfar, F., Devadas, S.: Physical unclonable functions and applications: A tutorial. Proceedings of the IEEE **102**(8), 1126–1141 (2014). https://doi.org/10.1109/JPROC.2014.2320516
10. Kang, H., Hori, Y., Satoh, A.: Performance evaluation of the first commercial PUF-embedded RFID. In: The 1st IEEE Global Conference on Consumer Electronics 2012. pp. 5–8 (2012). https://doi.org/10.1109/GCCE.2012.6379926
11. Knowm Inc.: Self directed channel memristors. rev. 3.2, October 6, 2019. Knowm, Santa Fe, NM, USA. `https://knowm.org/downloads/Knowm_Memristors.pdf`, Accessed: 2022-07-22
12. Kursawe, K., Sadeghi, A.R., Schellekens, D., Skoric, B., Tuyls, P.: Reconfigurable physical unclonable functions – Enabling technology for tamper-resistant storage. In: 2009 IEEE International Workshop on Hardware-Oriented Security and Trust. pp. 22–29 (2009). https://doi.org/10.1109/HST.2009.5225058
13. Menhorn, N.: External secure storage using the PUF. Application Note XAPP1333 (v1.2), April 12, 2022. Xilinx, San Jose, CA, USA. `https://www.xilinx.com/content/dam/xilinx/support/documents/application_notes/xapp1333-external-storage-puf.pdf`, Accessed: 2022-07-29
14. Najafi, F., Kaveh, M., Martín, D., Reza Mosavi, M.: Deep PUF: A highly reliable DRAM PUF-based authentication for IoT networks using deep convolutional neural networks. Sensors **21**(6) (2021). https://doi.org/10.3390/s21062009
15. Nozaki, Y., Shibagaki, K., Takemoto, S., Yoshikawa, M.: AI hardware oriented neural network physical unclonable function and its evaluation. Electronics and Communications in Japan **103**(11-12), 54–62 (2020). https://doi.org/10.1002/ecj.12276
16. Nugent, A.: Knowm memristor discovery manual. Knowm Inc. (September 2020)
17. Rose, G.S., McDonald, N., Yan, L.K., Wysocki, B.: A write-time based memristive PUF for hardware security applications. In: 2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). pp. 830–833. IEEE (2013). https://doi.org/10.1109/ICCAD.2013.6691209
18. Schaller, A., Xiong, W., Anagnostopoulos, N.A., Saleem, M.U., Gabmeyer, S., Škorić, B., Katzenbeisser, S., Szefer, J.: Decay-based DRAM PUFs in commodity devices. IEEE Transactions on Dependable and Secure Computing **16**(3), 462–475 (2019). https://doi.org/10.1109/TDSC.2018.2822298
19. Strukov, D.B., Snider, G.S., Stewart, D.R., Williams, R.S.: The missing memristor found. Nature **453**(7191), 80–83 (2008). https://doi.org/10.1038/nature06932
20. Uddin, M., Majumder, M.B., Beckmann, K., Manem, H., Alamgir, Z., Cady, N.C., Rose, G.S.: Design considerations for memristive crossbar physical unclonable functions. ACM Journal on Emerging Technologies in Computing Systems (JETC) **14**(1) (2017). https://doi.org/10.1145/3094414
21. Yilmaz, Y., Gunn, S.R., Halak, B.: Lightweight PUF-based authentication protocol for IoT devices. In: 2018 IEEE 3rd international verification and security workshop (IVSW). pp. 38–43. IEEE (2018)
22. Yin, C.E., Qu, G.: Temperature-aware cooperative ring oscillator PUF. In: 2009 IEEE International Workshop on Hardware-Oriented Security and Trust. pp. 36–42 (2009). https://doi.org/10.1109/HST.2009.5225055
23. Yue, M., Karimian, N., Yan, W., Anagnostopoulos, N.A., Tehranipoor, F.: DRAM-based authentication using deep convolutional neural networks. IEEE Consumer Electronics Magazine **10**(4), 8–17 (2021). https://doi.org/10.1109/MCE.2020.3002528
24. Zidan, M.A., Strachan, J.P., Lu, W.D.: The future of electronics based on memristive systems. Nature Electronics **1**, 22–29 (2018). https://doi.org/10.1038/s41928-017-0006-8