A graph matching algorithm for user authentication in data networks using image-based physical unclonable functions

Ali Valehi, Bertrand Cambou, Abolfazl Razi
School of Informatics, Computing and Cyber Security (SICCS)
Northern Arizona University, Flagstaff, AZ 86011
Emails:{ali_valehi, bertrand.cambou, abolfazl.razi}@nau.edu

***Abstract*—Recently, Physically Unclonable Functions (PUFs) received considerable attention in order to developing security mechanisms for applications such as Internet of Things (IoT) by exploiting the natural randomness in device-specific characteristics. This approach complements and improves the conventional security algorithms that are vulnerable to security attacks due to recent advances in computational technology and fully automated hacking systems.**

In this project, we propose a new authentication mechanism based on a specific implementation of PUF using metallic dendrites. Dendrites are nanomaterial devices that contain unique, complex and unclonable patterns (similar to human DNAs). We propose a method to process dendrite images. The proposed framework comprises several steps including denoising, skeletonizing, pruning and feature points extraction. The feature points are represented in terms of a tree-based weighted algorithm that converts the authentication problem to a graph matching problem. The test object is compared against a database of valid patterns using a novel algorithm to perform user identification and authentication. The proposed method demonstrates a high level of accuracy and a low computational complexity that grows linearly with the number of extracted points and database size. It also significantly reduces the required in-network storage capacity and communication rates to maintain database of users in large-scale networks.

I. Introduction

Image detection is commonly used as a non-secret identification method because cloning an image is hard to prevent, thereby the use of an additional secret password or a private key is important to ensure trustworthy authentication. The secret keys or passwords are commonly stored in non-volatile memory of secure microcontrollers such as electrically erasable programmable read-only memory (EEPROM), static random access memory (SRAM) and Flash memory. This approach is costly both in terms of key distribution and design procedure [1]. Moreover, security keys that are stored as digital information in memory are vulnerable to invasive attacks by adversaries who are constantly looking for ways to penetrate the security system and retrieve keys.

Utilizing Physically Unclonable Functions (PUF) is a new emerging security technique that has gained attention recently as a central building block in variety of cryptographic protocols. PUF technology leverages the natural randomness generated during the manufacturing of microelectronic components such as SRAM memories, ring oscillators, and gate delays. In this approach, a secret key which is required to produce a Challenge-Responses-Pair (CRP) for an authentication session is generated by PUF as a function of its hidden, hard-to-access and unique characteristics. A reference key is stored for each user in a secure databased in the network. An authentication process is deemed positive when the generated response by PUF matches the one in the network. In addition to the requirement of high randomness of keys generated by PUF to maximize the entropy of the cryptographic authentication process, PUFs should be resistant to side channel attacks. In this paper, we are investigating new methods to use the image of unclonning materials to produce secret PUFs that add an additional security level to the system and improve the current identification and trustworthy authentication methods.

As an implementation technology, we use a variant of nanomaterials based on metallic dendrites to generate an optical PUF. A sample dendrite object is depicted in Fig. 1. Metallic dendrites are formed by applying a constant voltage

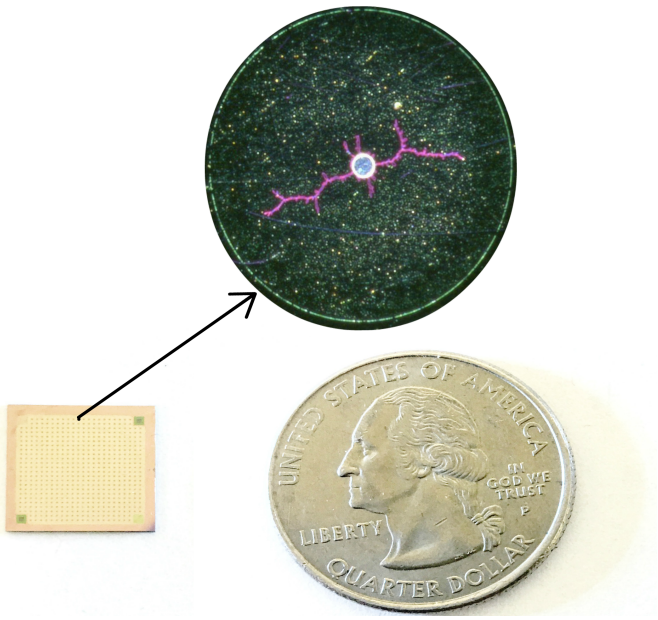

Fig. 1. A sample dendrite object. The dendrite wafer panel includes $24 \times 24 = 576$ dendrite objects.

to a localized metallic electrolyte which create random tree-like patterns [2], [3]. These devise-specific, hard-to-clone and extremely complex patterns are utilized as PUFs in this project to produce security signatures that are statistically unlikely to be extracted and reproduced by hackers. However, due to the extremely complex dendritic patterns, storing high-resolution images that preserves the pattern details as identification and authentication primitives in large-scale networks is practically infeasible. In this project, a novel method is proposed to develop representative graphs of dendritic patterns that capture the image security features to be used for user authentication. This mechanism can be used as an additional security stage in a wide range of data networks including wireless communication networks, Internet of Things (IoT), UAV networks and e-banking [1].

A. Image recognition techniques

The proposed research is closely related to the problem of *object recognition*, which is a well-studied problem in computer vision and image processing. Several algorithms are reported in the literature on image processing that deal with object recognition, which can be categorized to geometry-based approaches, appearance-based algorithms, and feature-based algorithms. The geometry-based approach is one of the earliest attempts for object recognition, which is based on finding a geometric representative of the object that models the representation variations in terms of Angle of View (AoV), illumination and other imaging parameters [4]. Appearance-based algorithms aim at capturing an object appearance using its main characteristic components. These algorithms are widely used to identify specific objects (e.g. faces) in an image [5]. Both geometry-based and appearance-based approaches require a large amount of information to store original high resolution images to ensure preserving sufficient details for an accurate identification with low false approval rates [4], [5].

Feature-based algorithms are considered to be the most popular and effective techniques, nowadays. In this approach, some description points are extracted from the object image as representative features of the image. These features are compared against a set of candidate object images and the one with a highest correlation is reported as the *best matched* object. Feature-based methods are desirable due to their high accuracies, robustness to imaging effects and low storage requirements [6].

There are two feature-based algorithms, which are arguably the most widely used techniques in computer vision applications. One is the Scale-Invariant Feature Transform (SIFT) algorithm that finds some local feature description points, which are invariant to scaling and orientation change [7]. The second method is called Speeded Up Robust Features (SURF) [8]. SURF is an algorithms inspired by SIFT including some modifications to reduce the computational complexity and provide a higher level of robustness against different image transformations. However, both techniques seem to be inappropriate for processing dendrite images and extracting security features due to several shortcomings. Firstly, these techniques are developed for general images with arbitrary patterns, hence are not customized for dendrite-specific tree-like pattern recognition. Secondly, the extracted feature points by these algorithms may not coincide with the dendrite pattern skeleton and hence are less useful for identification purpose as demonstrated in section IV.

In this paper, a novel algorithm is proposed for user authentication using centered tree-based dendritic patterns. The proposed method is based on mapping the extracted pattern skeletons to tree-based weighted graphs and using ideas from graph matching to find the matched object. This algorithm is shown to be computationally efficient, accurate and secure. Further, storing the representative graphs as surrogate images significantly reduces the storage capacity without considerable loss in capturing image features.

The rest of this paper is organized as follows. In section II, a brief review of related graph matching techniques is provided. In section III, the proposed method of converting the captured image into a weighted graph is proposed followed by a graph-matching algorithm to authenticate the test object. Section IV includes experimental results to verify the robustness of the proposed method in presence of various image distortions. Concluding remarks are provided in section V.

II. Review of graph matching algorithms

Graphs, due to their high representation powers, are used for structural description of objects in many research areas. Finding equivalence between graphs, the so called *graph isomorphism* is a powerful technique in computer vision that replaces the conventional correlation-based image similarity measuring methods [9].

Graph matching is known as an NP-Hard problem in general and a majority of heuristic-based algorithms are very slow with limited applicability. Another major issue with using graph matching for image recognition is the lack of robustness in dealing with image artifacts that result in missing actual pixels or inclusion of fake image pixels [10]. For instance, a majority of graph matching algorithms (such as greedy algorithms and labeling methods) that employ full graph structures for comparison purpose demonstrate exponential complexity with the number of nodes for worst-case scenarios [11]. Graph matching problem also can be cast as a nonlinear optimization problem. Using some sort of relaxation in labeling can significantly reduce the computational complexity compared to the exact labeling methods [12]–[14]. Another technique to reduce complexity is *subgraph matching* which is based on reducing the full graph matching to multiple smaller subgraphs matchings [15].

In this paper, a simplified method of subgraph matching for weighted graphs using Euclidean distance is proposed to evaluate similarity between a randomly generated pattern and a set of reference patterns. The proposed method is customized for tree-based graphs and presents a computational complexity

[1] A joint project is currently ongoing as a three university efforts (University of Arizona, Arizona State University and Northern Arizona University) supported by Arizona Board of Regents to develop next generation security devices based on Dendrites.

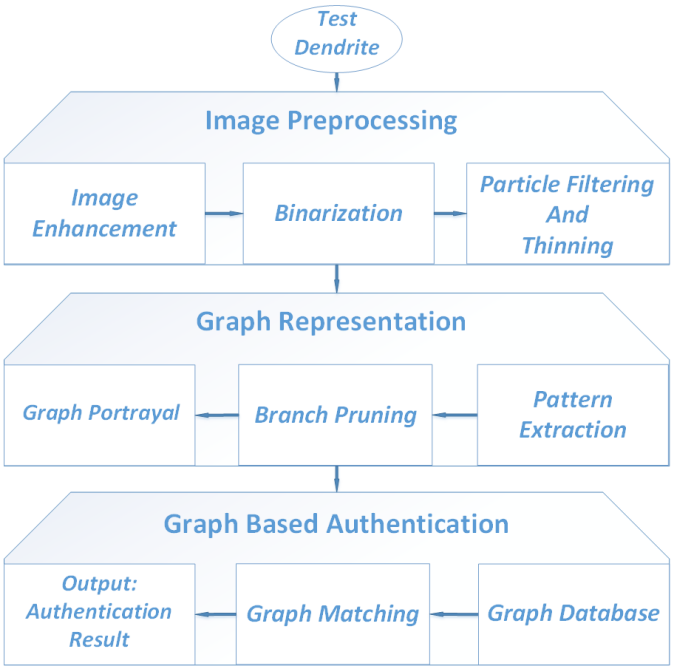

Fig. 2. System model: The proposed authentication framework includes i) image preprocessing, ii) graph representation and iii) graph-based authentication stages.

that grows linearly with the number of graph nodes, thereby is much faster compared to the other more general methods.

III. Methodology

In this section, a framework is proposed to recognize a legitimate network user by processing the image of embedded dendrite object and comparing it against a set of valid patterns. This framework comprises three main stages including *image preprocessing*, *graph representation* and *graph based authentication* as depicted in Fig. 2.

In the image preprocessing stage, the quality of captured image is improved to provide a reliable input for the subsequent stage of graph representation, where the enhanced image is processed and the main skeleton of pattern is extracted. The extracted pattern is then converted to a tree-based graph, where the tree intermediate and leaf nodes are corresponding to the skeleton bifurcation and end points, respectively. Further, a two-element vector is assigned to each edge that represents the Euclidean distance between the respective parent and child nodes as well as the angle of the branch with respect to a reference direction.

The final stage of the framework utilizes a graph matching algorithm to compare the developed graph for the test object with a set of valid graphs in the database in order to examine the validity and identity of the test object. The details of each stage are provided next.

A. Image Preprocessing

This stage processes the captured dendritic image through the following sequential steps:

Image enhancement: The quality of captured image may be influenced by random noises caused by environment light

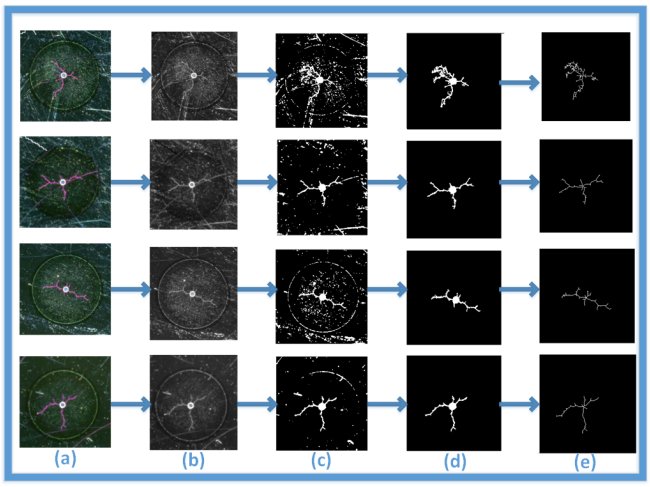

Fig. 3. Illustration of image preprocessing stage. Each row is corresponding to a dendrite object. The columns are outcomes of different algorithm steps including (a) image enhancement, (b) grayscaling (c) binarization (d) particle filtering and (e) thinning.

intensity variation, equipment noise, and light reflections. Noise parameters of the image are estimated using a patch-based noise level estimation techniques proposed in [16]–[18]. An efficient implementation of this technique is publicly available in [19], which is used in this work (Fig. 3(a)). Further, printing a common pattern such as a circle or a cross (†) with a predefined structure on the image can be used to compensate the undesired effects of image rotation, scaling and skewness.

Binarization: The next step is to convert the enhanced image to a grayscale image (Fig. 3(b)) and then to a monochrome image using a predefined threshold (Fig. 3(c)).

Particle filtering: It is noteworthy that image denoising enhances the quality of the image, but is not capable of fully eliminating all image artifacts. Some of the remained artifacts may add additional fake lines to the image that significantly reduce the accuracy of the proposed algorithm. Noting the fact that dendrite branches form tree-like patterns, all disconnected lines and points are considered fake. Therefore, a particle filtering is utilized to smooth the image and exclude disconnected noisy particles (Fig. 3(d)).

Thinning: Finally, in order to obtain a coherent skeleton, pixels on the boundaries of the skeleton are removed until the object thickness become one pixel (Fig. 3(e)).

B. Graph Representation

Pattern extraction: The obtained image skeleton is processed in this stage to extract the dendrite unique pattern. Here, we take advantage of a common feature shared across all objects, which is an equi-size circle centered in the middle of the device as shown in Fig. 3. Center of this reference circle identifies the starting region for the pattern extraction process. In this stage, we first detect the circle using Hough transform, which is a feature extraction technique to detect a classified shape in an image [20]. Once the reference circle is detected, the center point and the radius are readily obtained. The branched spreading out from the circle perimeter identify the seed points to develop the rest of the tree.

For a monochrome image, it is sufficient to track the connected paths of black pixels appear on the white background in order to extract the whole fully connected pattern. We slide a 3×3 scanning window starting from the seed points and following the branch lines. The window is moved only a single pixel at each step and the direction of the motion is aligned with the branch direction.

There are different scenarios based on the part of the pattern that the sliding window visits during its journey along the branches, as depicted in Fig. 4. The scenarios include visiting seed points, bifurcation points, regular points (intermediate pixels on the branches), and end points. Different pixel types can be identified using the following set of simple rules.

We first represent the black and white pixels with digits '1' and '0', respectively. The 3×3 scanning window includes 9 pixels, each of which is labeled with a number between 1 and 9 as shown in Fig. 4. After sliding the window to the new position, we form the following sets:

$$\begin{cases} \mathcal{A} : \text{set of black pixels ('1') scanned in the previous position,} \\ \mathcal{B} : \text{set of all black pixels ('1') in the current position,} \\ \mathcal{I} : \mathcal{B} \setminus \mathcal{A} \end{cases} \tag{1}$$

The pixels in $\mathcal{A}$ and $\mathcal{B}$ are shown by dotted and colored squares in Fig. 4, respectively. $\mathcal{I} = \mathcal{B} \setminus \mathcal{A}$ includes the newly visited points and defines the currently visited pixel type as follows:

For each new position of the window, if there are two or more newly visited '1' pixels, a bifurcation pixel is detected. For example in Fig. 4(b), suppose that the window is moved from the previous center point 7 to the new center point 5. This means that the black pixels in blocks 7 and 5 are already visited in the previous position, when the 3×3 window was centered at block 5 ($\mathcal{A} = \{7, 5\}$). In the current position, the black pixels form the set $\mathcal{B} = \{1, 5, 7, 9\}$. Therefore, the set $\mathcal{I} = \mathcal{B} \setminus \mathcal{A} = \{1, 9\}$ includes two new pixels 1 and 9 and consequently represent a bifurcation point. In such a case, branch tracking is continued in both directions defined by the new nodes 1 and 9.

Regular pixels are intermediate points on the branch lines. This type of pixel is detected when there is only one newly visited black pixel in the current position. For instance, Fig. 4(c), represents a scenario where the window is moved from center point 8 to the new position at 5. In this scenario, we have $\mathcal{A} = \{8, 5\}$ and $\mathcal{B} = \{3, 5, 8\}$. Consequently $\mathcal{I} = \mathcal{B} \setminus \mathcal{A} = \{3\}$ includes only one newly visited pixel (3) and this part of pattern is considered as a regular point. Regular points are not used in tree formation and are only used to direct the window to the new position.

Finally, if no new black pixel is detected ($\mathcal{I} = \{\}$) as shown in Fig. 4(d), it is considered an end point. These rules are summarized in (2).

$$|\mathcal{I}| = \begin{cases} 0: & \text{Seed Point or End Point} \\ 1: & \text{Regular Point} \\ \geq 2: & \text{Bifurcation Point} \end{cases} \tag{2}$$

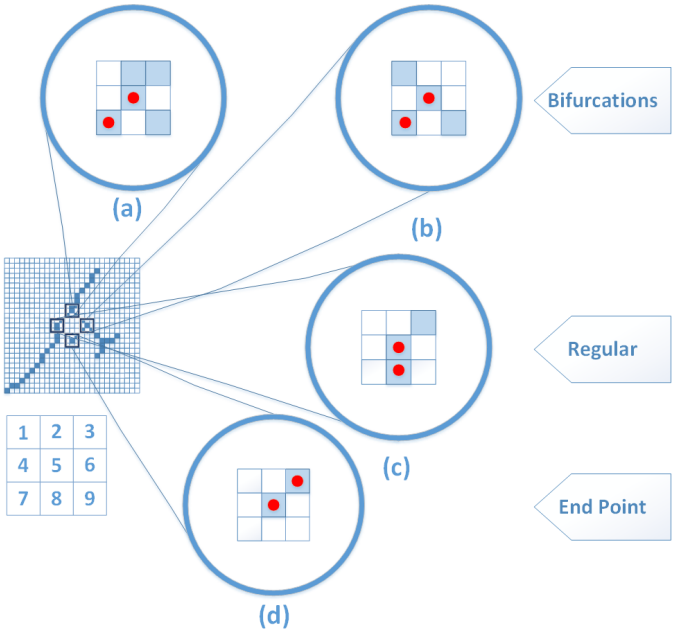

Fig. 4. Different point types met by a 3×3 scanning window along the skeleton. (a,b) bifurcation, (c) regular, (d) endpoint.

Branch pruning: As mentioned in section III-A, some noise and distortion effects are not fully eliminated in the denoising stage. This can increase the rate of false identification of bifurcation points. For instance, a Gaussian noise can make the image blurred that may misrepresent a single branch as two separated branches in the skeletonization process. Therefore, we eliminate the suspected branches in order to reduce false bifurcation point report rate. The deletion criterion here is based on connectivity and branch length. We eliminate the disconnected branches as well as the branches that connect children nodes with distances less than a predefined threshold to their parent nodes.

Graph portrayal: In this step, we eliminate the regular points and consider only the seed points, bifurcation and end points in order to form the representative tree. The middle point of the object (defined by the center of the reference circle) is assumed to be the root note of the tree. The seed points are considered the first-level children nodes. The further bifurcation nodes are the next level descendant nodes. Finally, the end points are corresponding to the tree leaves. For each edge connecting the node n_i to its parent $\mathcal{P}(n_i)$, we assign a vector $\alpha_i = (d_i, \theta_i)$, where $d_i = d\big(n_i, \mathcal{P}(n_i)\big)$ is the Euclidean distance between n_i and $\mathcal{P}(n_i)$. θ_i is the angle between the line connecting n_i to $\mathcal{P}(n_i)$ with respect to an arbitrarily chosen reference direction.

C. Graph-Based Authentication

Here, we present an algorithm to authenticate a test dendrite object by comparing its representative graph to a set of valid reference graphs supposedly stored in a database in the network. Each reference graph in the database is assigned with a unique ID and is corresponding to a unique dendrite object. The algorithm checks the test graph and reports the ID of a

most similar reference graph. If the reported ID matches the test device ID, the authentication is successful.

The proposed algorithm is based on measuring similarity between the test and reference objects. The similarity measure is evaluated in a sequential manner starting from the first layer of tree up to the very last layer. At each comparison level, the reference graphs are sorted based on their similarity to the test object. Moreover, the search space is reduced at each comparison level by filtering out the least similar objects. The test object that ranks first at the last comparison level is reported as the *best matched* object provided that it satisfies a minimum similarity criterion.

Before elaborating on different steps of this algorithm for each comparison level, we describe the used notation. We denote the test graph with T and the set of M reference graphs with $\mathcal{R} = \{R_1, R_2, \ldots, R_M\}$. The level of a node n denoted by $l(n)$ is the minimum number of edges connecting the node to the root node. We have $l \in \{1, 2, ..., L\}$, where L is the number of levels. $\mathcal{N}(l)$ is the set of nodes in level l. The sibling set for node n is defined as $\mathcal{S}(n) = \{m | \mathcal{P}(m) = \mathcal{P}(n)\}$ and includes the nodes that share the same parent with node n, where $\mathcal{P}(n)$ denotes the parent of n. We also define self-sibling set $\mathcal{H}(n) = \mathcal{S}(n) \cup \{n\}$ as the set of sibling nodes of n including the node n itself. Now, we proceed with detailing the algorithm steps for comparison procedure at level $l \in \{2, 3, \ldots, L\}$:

Step 1 (self-sibling set formation): At level l, we first develop k_l disjoint self-sibling sets $\mathcal{H}_i^{(l)}, i \in \{1, 2, \ldots, k_l\}$ that partition $\mathcal{N}(l)$, the set of .... of all nodes at level l. Therefore, we have $\mathcal{H}_i^{(l)} \cap \mathcal{H}_j^{(l)} = \{\}$ for $\forall i, j \in \{1, 2, \ldots, k_l\}$ and $\mathcal{H}_1^{(l)} \cup \mathcal{H}_2^{(l)} \cup \cdots \cup \mathcal{H}_k^{(l)} = \mathcal{N}(l)$. We repeat this procedure for the test graph T and all reference graphs $R_1, R_2, \ldots, R_M$.

It is noteworthy that at level $l = 2$, there is only one self-sibling set $\mathcal{H}_1^{(2)}$, since all nodes at this level share the same parent (root node). However, for comparison levels $l > 2$, there are more than one self-sibling sets. In this case, a mapping between these sets is already established in comparison level $l - 1$ (as we explain next).

Step 2 (distance evaluation): Here, we generate a distance matrix for all nodes in a self-sibling subset $\mathcal{H}_j^{(l)}$ between the test graph and references graph R_i, denoted by $W_j^{(l)}(T, R_i) = \{d(n, m) | n \in \mathcal{H}_j^{(l)}(T), \ m \in \mathcal{H}_j^{(l)}(R_i)\}$. We arbitrarily use Euclidean distance defined as $d(n, m) = \sqrt{\alpha(\theta_n - \theta_m)^2 + (1 - \alpha)(d_n - d_m)^2}$, where α is a tunning parameter. We repeat this procedure for all self-sibling sets mapped between the test graph and all reference graphs.

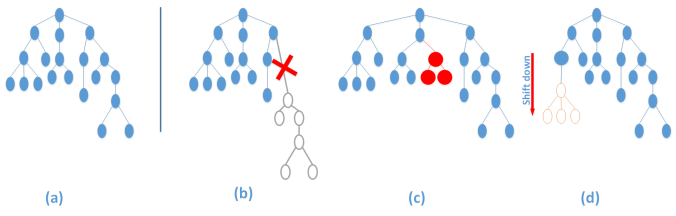

Fig. 5. Impact of noise on subgraph mapping. a) a subgraph is missed due to missing a black intermediate pixel in pattern extraction module. b) insertion of a fake subgraph due to noise. c) level change of a subgraph due to false identification of a junction node.

Step 3 (mapping): Once, we have distance metrics between all possible pairs of nodes belonging to a particular self-sibling set (in the test graph and each of reference graphs), we find an optimal mapping between the nodes within a self-sibling subset. Optimal mapping minimizes the overall pair-wise distances among all possible pairs. We use an algorithm called *Munkres*, which is a combinatorial optimization algorithm to construct a maximum weight perfect matching in bipartite graphs [21]. After running this step for all self-sibling sets, a mapping $\Pi_i^{(l)}$ is established for all nodes at level l between the test graph T and the reference graph R_i. It is noteworthy that the root nodes of reference graph and test graph are always mapped together. This mapping is used in the next comparison level to map the self-sibling sets that emerge from these nodes. We repeat this procedure for all reference graphs $R_1, R_2, \ldots, R_M$.

Step 4 (distance averaging): In this step, we calculate distances between the test graph and the reference graphs R_i simply by average the obtained pairwise distances across all nodes defined by mapping $\Pi_i^{(l)}$ (i.e. $D_i^{(l)}(T, R_i) = \frac{1}{|\mathcal{N}(l)|} \sum_{(n,m) \in \Pi^{(l)}} d(n, m)$). We repeat this procedure for all reference graphs.

Step 5 (ranking): At this step, we rank the graphs based on their average distances from the test object. After ranking, we eliminate the least similar reference objects. Here we choose to remove $\beta = 0.8$ of the available graphs at each level.

Step 6 (termination criteria check): The last step is to check the termination criteria by examining the most similar reference graph. If the distance from the test object is above a predefined threshold γ and the distance for this reference object is at least 20% lower than that of the second most similar graph, we report this reference object as the *best matched* and terminate the program. The second condition is to ensure that the best match is a true match with a considerable distinction with other potential candidates. Otherwise, we proceed with the next level until all levels are evaluated.

Algorithm performance: If the ID of the identified *best matched* graph is equal to the that of the test graph, the authentication is successful, otherwise the authentication fails (mismatch error). If a reference object is reported as the *best match* for an invalid test object without actual ID in the reference database, the event is considered as false positive. Also, if no reference graph meets the similarity criteria for a test object with a valid ID, it is considered as false negative. These three error types quantifies the error rate of the proposed algorithm in section IV.

Remark: Noise and other undesired distortion effects can insert fake tree nodes or remove true valid nodes and hence negatively impact the mappings between the nodes at each comparison level, as shown in Fig. 5. We modify the above algorithm slightly to accommodate this issue.

For instance, missing a pixel from a branch in the pattern may cause missing a graph edge and in turn missing its downstream subnetwork as shown in Fig. 5(b). Likewise, a

fake branch may appear in the graph as shown in Fig. 5(c). These two types of incorrect configurations can be detected and solved as follows. When the numbers of nodes at the same level of test and reference graphs are not equal, we suspect that a branch is missing either in the test or the reference graph. Thereby, mapping is performed only among the existing nodes. Therefore, additional nodes and their downstream subnetworks are considered as fake nodes and are eliminated from the further comparison levels. Moreover, a predefined penalty is assigned to a graph after each occurrence of mismatch in the number of nodes at the same level comparison.

In addition to the aforementioned scenarios, insertion or deletion of a bifurcation node may cause a level change for a subgraph that in turn disrupts the mapping procedure. Such a case is seen in Fig. 5(d), where inclusion of a fake node at level 2 shifts the corresponding subgraph down to the 3rd level. To solve this issue, we propose to compare the subtrees (including a node and its direct children) instead of individual nodes when evaluating distances between two nodes at the same level. More specifically, for each node of the test graph, we consider its first generation children and itself as a new graph (subgraph A) and a family of compared nodes in the reference graph as another graph (Subgraph B). Then these two graphs are compared together. As shown in Fig. 6, the distances between the root node of subgraph A with respect to every node of subgraph B are calculated. Then, the nearest node is mapped to the root node of subgraph A. This modification significantly improves the performance of the proposed algorithm by adjusting the misplaced subgraphs.

Computational complexity: The proposed graph matching algorithm can complete the steps in $O(NM)$ at worst case, where M is the number of reference graphs and N is the number of nodes in the test graph. The simple justification is that in the proposed algorithm, the relative distance of each node in the test graph is calculated only with respect to a few candidate nodes belonging to the same self-sibling subset (in most cases to 2 candidate nodes). Therefore, the complexity grows linearly with the number of nodes. Apparently, the complexity grows linearly with the number of test objects as well, since the algorithms is repeated for each test graph. Note that there is a sort operation between reference object at each level l. The number of levels are proportional to $\log(N)$. The comparison complexity for M object using standard sort methods such as *Heap sort* is $M\ \log(M)$. Therefore, the sort complexity at worst case (considering M object at each level and not eliminating less similar objects) is $O(M\ \log(NM))$, which is negligible compared to $O(MN)$ for $N \ll M$.

IV. Experimental result

In order to verify the accuracy and robustness of the proposed algorithm, we applied it to a set of test dendrite objects. At each experiment, the reference database includes 25 reference objects. We applied the algorithm on 10 test objects and the average results are presented. We applied different levels of Gaussian noise and skewness to simulate imaging artifacts in real environments.

Algorithm 1: Graph-matching based authentication

Input: Test graph (T), a set of reference graphs ($\mathcal{R} = \{R_1, R_2, ..., R_M\}$).
Parameters: Number of levels (L), desired similarity threshold (γ), distance tunning parameter (α), dissimilar graph elimination factor (β), reliability factor (δ)
Output: -Validation result (device is valid or not?)
-ID of best matched object
begin
  for $l \leftarrow 2$ **to** L **do**
    for $R_i \in \mathcal{R}$ **do**
      Set $H^{(l)}(T) = \{H_1^{(l)}, ...H_{k_l}^{(l)}\}$ (self-sibling sets of test graph at level l)
      for $\mathcal{H}_j^{(l)} \in \mathcal{H}^{(l)}(T)$ **do**
        $\mathcal{H}_j^{(l)}(R_i) \leftarrow$disjoint self-sibling of R_i at level l according to $\Pi_i^{(l-1)}$.
        –Generate $W_j^{(l)}(T, R_i)$ (The distance matrix for all nodes in $\mathcal{H}_j^{(l)}(R_i)$ and $\mathcal{H}_j^{(l)}(T)$).
      –$\Pi_i^{(l)} \leftarrow$ optimal mapping between $\mathcal{H}^{(l)}(T)$ and $\mathcal{H}^{(l)}(R_i)$ according to $W_j^{(l)}$.
      –$D_i^{(l)} \leftarrow$ average over the obtained pairwise distances across all nodes defined by $\Pi_i^{(l)}$.
    –Sort $\mathcal{R}$ according to $D_i^{(l)}$ ascendingly (R_1 has smallest value).
    –Eliminate last β percentage of the remaining members of $\mathcal{R}$.
    –$D_i'^{(l)} \leftarrow D^{(l)}$ of sorted $\mathcal{R}$.
    if $((D_1'^{(l)} > \gamma)$ *and* $((\delta)(D_2'^{(l)}) >= D_1'^{(l)}))$ *or* $(l = L)$ **then**
      –Report R_1 as the best matched.
      Exit for.
    else
      –Report T as an unidentified graph.

Fig.7 compares the size of files required to store the original images (in PNG compressed format) with the files used to store representative graphs (in MAT format) for different number of objects. It is seen that the required storage capacity is approximately reduced by a factor of 100 with respect to the compressed image. This considerable gain does not degrade the algorithm accuracy, since the preserved points (tree nodes) provide sufficient information for reliable authentication. Therefore, the proposed method is very efficient for large-scale networks. In fact, storing the original images for all users in a large-scale network is almost infeasible due to the large storage and high communication rate requirements to store and update the reference database.

In practical applications, the common issue of camera misplacement can change the angle of view (AoV) in imaging, which imposes distortion on the image such that different branches experience different scaling (skewness effect). Also,

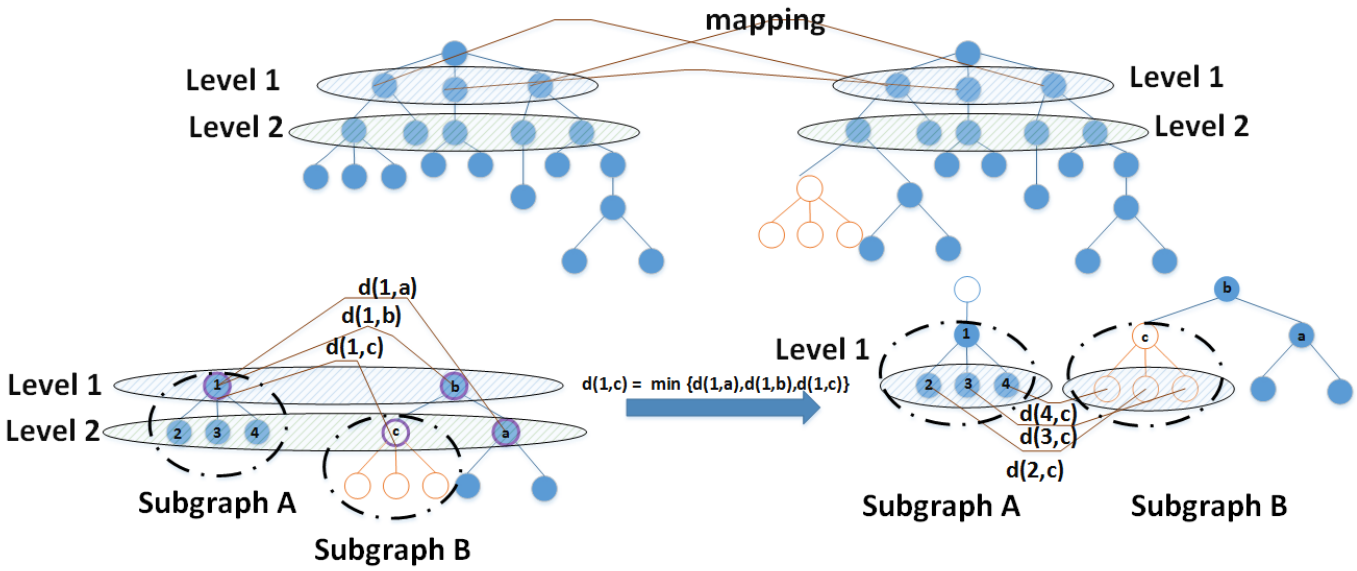

Fig. 6. Demonstration of the proposed algorithm to find similarity between the test graph and a set of reference graphs. Top row presents sub graph mapping stage. The bottom row, represents identification and recovery of level change by the proposed modified algorithm.

the branch directions and angles may vary slightly. This phenomenon considerably deteriorates the performance of correlation based methods. However, since the proposed algorithm extracts the whole structure of the pattern in terms of relative distances and angles it is not sensitive to these minor variations and demonstrates negligible performance degradations.

To examine the proposed algorithm we applied skewness up to 12 degrees on 10 different test objects. Then relative distance of the skewed image from each of the reference objects were calculated. As it can be seen in Fig. 8, the test image even after skew effect up to 11 degrees represents a less distance to the original unskewed reference image in comparison with other reference objects. In addition, calculated distances satisfy the termination criteria. In other words, a skewness up to 11 degrees has no significant effect on the algorithm accuracy.

Another key advantage of the proposed algorithm with respect to the benchmark feature based methods such as SIFT and SURF is extracting feature points that collectively form the structure of the unique pattern of the test object. This is in clear contrast with other methods that choose feature points potentially from the background noise. This phenomenon is depicted in Fig. 9. In fact, the simulation results suggest that SIFT and SURF (and perhaps other similar algorithms) fail in capturing the pattern structure by not choosing appropriate feature points and thereby perform poorly in the image authentication.

Finally, Fig.10 presents two important performance metrics (sensitivity and specificity) of the proposed method by varying the similarity criterion threshold (γ). Sensitivity is defined by the successful verification rate of valid objects. whereas specificity is defined by falsely recognizing invalid objects. The test is performed using 10 fake objects with no valid ID in in database and 10 objects under Gaussian noise with mean value equal to zero and variance equal to 0.02. As expected, using a loose threshold value improves the algorithm

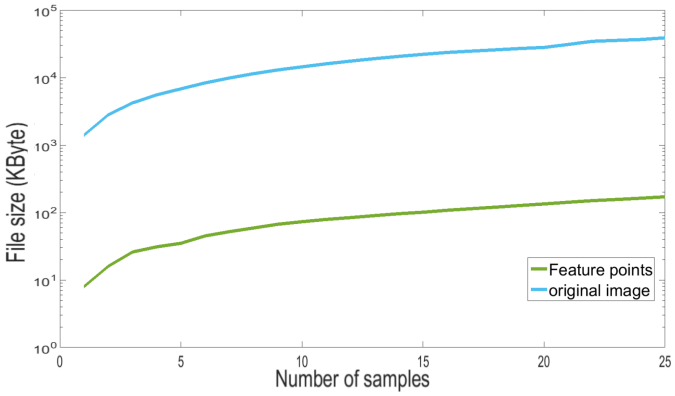

Fig. 7. Comparison of file sizes required to store the original image file size (PNG format) and the extracted graphs (in MAT formats).

sensitivity by reducing false negative rate. The loose threshold level also increases the number of false positive rate (poor specificity). Therefore, the threshold level should be chosen carefully to make a balance between the specificity and sensitivity based on the system requirements. The region under the ROC curve is approximately (0.90), that demonstrates a desirable performance level.

V. Conclusion

In this paper, a new authentication framework is proposed based on dendrite objects as a new implementation of PUF technology. The unique and unclonable patterns of these nanomaterials are used to generate security primitives that are reliable and barely breakable. The proposed approach converts the image recognition problem to a graph matching problem with a novel solution that accommodates subgraph level changes. The proposed method reduces storage and communication rate requirements by a factor of 100, which makes it well-suited for next generation data networks such as Internet of Things. The proposed algorithm is highly robust to distortion and

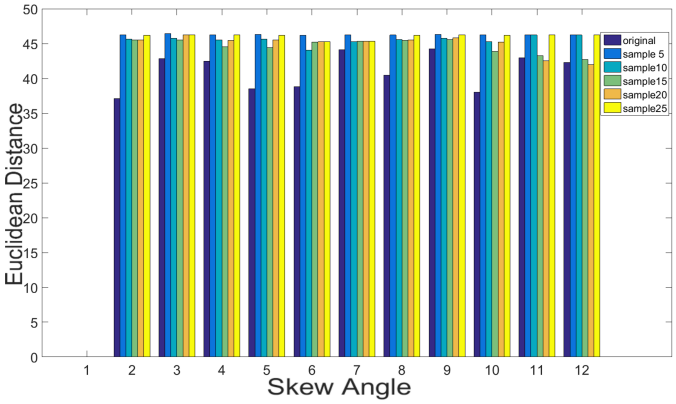

Fig. 8. Distances between the representative graph of a skewed image with 25 reference sample graphs in the database

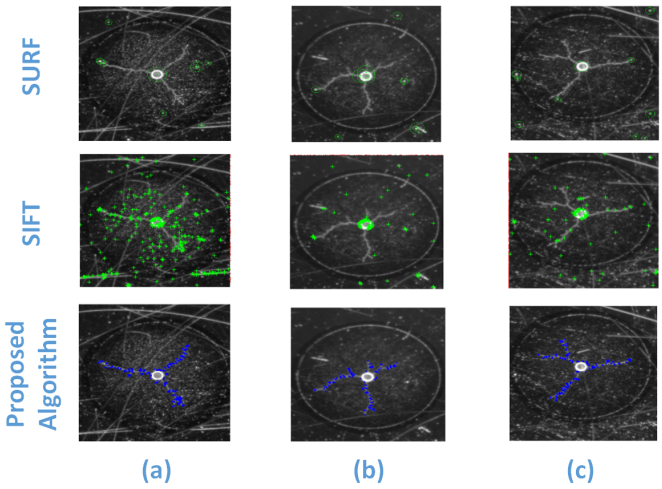

Fig. 9. Comparison between the extracted feature points using SURF, SIFT and the proposed algorithm. Each column is corresponding to a different sample object.

skew effects and outperforms the classical image recognition and graph matching algorithms by considering the specific properties of tree-like dendrite patterns. Using this technology, adds an additional security level to the conventional data-based security algorithms that become more and more vulnerable to security attacks.

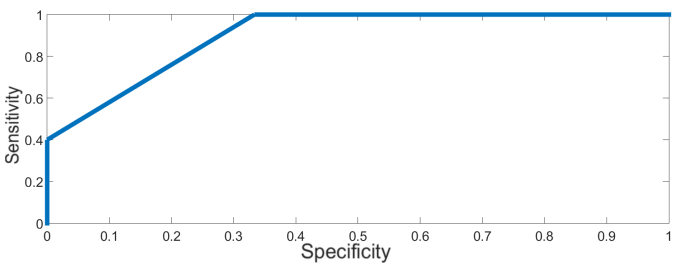

Fig. 10. ROC curve: The trade off between the sensitivity and specificity of the proposed authentication method by varying termination criterion parameter (γ).

ACKNOWLEDGMENT

The authors would like to thank Professor Michael Kozicki for providing dendrite samples and helpful comments on extracting security primitives from dendritic patterns.

REFERENCES

[1] C. Herder, M. D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126–1141, 2014.

[2]

[3] M. Kozicki and M. Ren, "Dendritic metal structures, methods for making dendritic metal structures, and devices including them," July 2015. [Online]. Available: https://www.google.com/patents/US20150194545

[4] J. Mundy and A. Zisserman, "Geometric invariance in computer vision," *MIT Press*, 1992.

[5] X. Liu, A. Srivastava, and K. Gallivan, "Optimal linear representations of images for object recognition," *TPAMI*, vol. 26, p. 662666, 2004.

[6] J. W. Howarth, H. H. C. Bakker, and R. C. Flemmer, "Feature-based object recognition," in *Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on*, Feb 2009, pp. 375–379.

[7] D. Lowe, "Object recognition from local scale-invariant features." *Proceedings of International Conference on Computer Vision (ICCV99)*, vol. 102, 1999.

[8] H.Bay, A. Ess, T.Tuytelaars, and L. Gool, "Speeded up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, p. 346359, 2008.

[9] A. Shokoufandeh and S. Dickinson, "Theoretical aspects of computer science," G. B. Khosrovshahi, A. Shokoufandeh, and A. Shokrollahi, Eds. New York, NY, USA: Springer-Verlag New York, Inc., 2002, ch. Graph-theoretical Methods in Computer Vision, pp. 148–174. [Online]. Available: http://dl.acm.org/citation.cfm?id=644608.644614

[10] S. Gold and A. Rangarajan, "Graph matching by graduated assignment," in *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, Jun 1996, pp. 239–244.

[11] B. Gallagher, "Matching structure and semantics: A survey on graph-based pattern matching," *AAAI FS*, vol. 6, pp. 45–53, 2006.

[12] A. Rosenfeld, R. A. Hummel, and S. W. Zucker, "Scene labeling by relaxation operations," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 6, pp. 420–433, June 1976.

[13] L. S. Davis, "Shape matching using relaxation techniques," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 1, pp. 60–72, Jan 1979.

[14] R. A. Hummel and S. W. Zucker, "On the foundations of relaxation labeling processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no. 3, pp. 267–287, May 1983.

[15] D. Eppstein, "Subgraph isomorphism in planar graphs and related problems," in *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '95. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1995, pp. 632–640. [Online]. Available: http://dl.acm.org/citation.cfm?id=313651.313830

[16] X. Zhu and P. Milanfar, "Automatic parameter selection for denoising algorithms using a no-reference measure of image content," *IEEE Transactions on Image Processing*, vol. 19, no. 12, pp. 3116–3132, Dec 2010.

[17] X. Liu, M. Tanaka, and M. Okutomi, "Noise level estimation using weak textured patches of a single noisy image," in *2012 19th IEEE International Conference on Image Processing*, Sept 2012, pp. 665–668.

[18] ——, "Single-image noise level estimation for blind denoising," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5226–5237, Dec 2013.

[19] A. MESHRAM. (2014) Noise level estimation. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/45940-noise-level-estimation/content/NLEstimate.m

[20] J. Illingworth and J. Kittler, "The adaptive hough transform," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 5, pp. 690–698, May 1987. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.1987.4767964

[21] Y. Cao. (2008) Hungarian algorithm for linear assignment problems (v2.3). [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/20652-hungarian-algorithm-for-linear-assignment-problems–v2-3-/content/munkres.m