Protecting an Intrusion Detection Engine that Uses Neural Networks with Watermarking Techniques in Mobile Ad Hoc Networks

Aikaterini Mitrokotsa[1], Nikos Komninos[2], Christos Douligeris[1]

[1] Department of Informatics, University of Piraeus,
80 Karaoli & Dimitriou Str. Piraeus, 18534, Greece
{mitrokat,cdoulig}@unipi.gr

[2] Athens Information Technology,
19002 Peania Attiki, Greece
nkom@ieee.org

Abstract. Mobile ad hoc networks have received great attention in recent years, mainly due to the evolution of wireless networking and mobile computing hardware. Nevertheless, many inherent vulnerabilities exist in mobile ad hoc networks and their applications that affect the security of wireless transactions. As intrusion prevention mechanisms, such as encryption and authentication, are not sufficient we need a second line of defense, Intrusion Detection. In this paper we present an intrusion detection engine based on neural network**s** and a protection method based on watermarking techniques. In particular, we exploit information visualization and machine learning techniques in order to achieve intrusion detection and we authenticate the maps produced by the application of the intelligent techniques using a novel combined watermarking embedding method. The performance of the proposed model is evaluated under different traffic conditions, mobility patterns and visualization metrics.

Keywords: Intrusion Detection, Neural Networks, Mobile ad hoc networks, Watermarking

1. Introduction

Wireless communication is gaining adoption in a broad range of environments making essential the need for rapid proliferation of wireless networking technologies. Mobile ad hoc networks (MANET), also called spontaneous networks, could be defined as a collection of mobile nodes, which employ a multi-hop information transfer without relying in an a-priori infrastructure. All nodes communicate in a self-organized way and are able to appear and disappear from the network at any time. Mobile devices create a wireless communication channel and each of them contributes

in the routing decisions of the network and the basic network services. Mobile nodes communicate directly with nodes in their vicinity and use intermediate nodes in order to exchange information with nodes out of their radio range. Cooperation is a substantial requirement for the effective performance of the wireless ad hoc network. Because of the special advantages that wireless ad hoc networks present, they are envisioned in many critical applications including battlefields and disaster recovery applications.

Although MANETs are characterised by great flexibility, they also present many inherent vulnerabilities that pose essential research challenges and unique security requirements. MANETs are characterized by a dynamically changing topology leading to a not well-defined boundary where access control mechanisms and firewalls can be applied. A MANET is susceptible to numerous threats including passive eavesdropping, spoofing and modification of information. Their vulnerability is further stressed by nodal interdependency and resource constraints including limited battery, bandwidth and CPU use.

Intrusion Detection is an invaluable mature arsenal with a long history of research for the defense of wired networks but is still in its infancy in the area of MANETs. In this paper we present an intrusion detection module that is part of a local IDS architecture composed of a data collection engine, an intrusion detection engine and a response engine. The focus of this paper is the proposal of the intrusion detection module that is based on a type of neural networks known as emergent Self-Organizing Maps (eSOMs). Neural Networks have the great advantage of tolerance towards imprecise data. We exploit this important feature of neural networks in order to classify normal against abnormal behavior in MANETs. Combining machine learning, information visualization and watermarking techniques we are able to have a clear view of how secure a MANET is against attacks. In particular, each node of the MANET creates a map that depicts its security state and distributes this map to all its neighboring nodes. Thus, each node knows the security status of every neighbor by generating a global map. The global map is used in order to perform secure and efficient routing by avoiding paths that include nodes, which are victims of attacks.

Furthermore, watermarking techniques are applied in order to protect the produced maps from modification. The proposed intrusion detection approach uses a combined watermarking technique that derives from Lattice and Block-Wise ([1], [2], [3]) methods. When local maps are broadcasted to the neighboring nodes, a cryptographic encoder and decoder can authenticate them.

Neural Networks and watermarking techniques are two research areas with many advantages that have never been used before in the research area of MANET. We exploit their main advantages and we propose an intrusion detection module, which is part of a local IDS agent. Using eSOM, we to have a visual representation of the normal-attack state in each node of the MANET. Hence, each node can determine whether a neighboring node is under attack and forward its messages accordingly. Moreover, with watermarking techniques we can authenticate the nodes of the MANET and verify the integrity of the maps produced by eSOM.

Following this introduction, the paper is organized as follows. Section 2 presents related work of intrusion detection approaches that have been proposed for mobile ad hoc networks and approaches that use watermarking techniques. Section 3 discusses the intrusion detection model this paper is based on. Section 4 presents a functional

description of the proposed detection engine and the classification algorithm used. Section 5 presents the watermarking technique proposed for the authentication of maps produced by eSOMs. In section 6 the performance evaluation of the detection engine as well as the results of the proposed watermarking technique are presented. Finally section 7 concludes the paper and discusses some future work.

2. Related Work

Intrusion Detection is an active and mature research area for wired networks but techniques designed for wired networks may not be efficient if applied to wireless ad hoc networks due to the stringent requirements these networks present. Compared with wired networks where traffic monitoring is performed in gateways, routers and switches, wireless ad hoc network lack centralized choke points at which it would be possible to monitor network traffic. Even if we could achieve the existence of such concentration points, their locations would continuously change due to mobility. That is why the deployment of a distributed intrusion detection approach in wireless ad hoc networks is a necessity. Additionally, we should focus on security mechanisms keeping in mind the ease of listening to wireless transactions, the lack of a fixed infrastructure and the resource consumption characteristics of MANET. This means that it is better to use a periodic intrusion detection system (IDS) than an 'always-on' prevention mechanism. Moreover, the resource constraints that MANETs face including limited battery, bandwidth and frequent miscommunication complicate the discrimination between a new qualified operation after a disconnection and an intrusion. Something that makes even more difficult the classification between normal and anomaly behavior.

Zhang and Lee [4] proposed the first (high-level) IDS approach specific for ad hoc networks. They proposed a distributed and cooperative anomaly based IDS, which provides an efficient guide for the design of IDS in wireless ad hoc networks. They focused on an anomaly detection approach based on routing updates, on the MAC layer and on the mobile application layer.

Huang and Lee [5] extended their previous work by proposing a cluster-based IDS, in order to combat the resource constraints that MANETs face. They use a set of statistical features that can be derived from routing tables and apply the classification decision tree induction algorithm C 4.5 in order to detect normal vs. abnormal behavior. The proposed system is able to identify the source of the attack, if the identified attack occurs within one hop.

Deng et al. [6] proposed a hierarchically distributed and a completely distributed intrusion detection approach. The intrusion detection approach used in both of these architectures focuses on the network layer and it is based on a Support Vector Machines (SVM) classification algorithm. They use a set of parameters derived from the network layer and suggest that a hierarchically distributed approach may be a more promising solution versus a completely distributed intrusion detection approach.

Kachirski and Guha [7] proposed a cluster-based intrusion detection system built on a mobile agent framework. The proposed system uses mobile agents each perform-

ing a particular role, either monitoring, or decision or action. A few nodes are chosen by a distributed algorithm in order to host sensors for the monitoring of network packets and agents in order to make the decisions. Additionally, all the nodes host sensors for host-based monitoring. The main advantage of this approach is that the packet-monitoring task is limited in a few nodes and the IDS-related processing time by each node is minimized.

Liu et al. [8] proposed a completely distributed anomaly detection approach. They investigated the use of the MAC layer in order to profile normal behavior of mobile nodes and then apply cross-feature analysis [9] on feature vectors constructed from the training data.

Tseng et al. [10] proposed a distributed and specification based intrusion detection approach in order to detect attacks in the AODV routing protocol. The approach involves the use of finite state machines. More specifically correct AODV routing behavior is specified using finite state machines and the actual behavior of AODV flows is compared with these specifications. Any deviation from these specifications is recognized as intrusion. Specification based techniques have the drawback that it is necessary to balance the tradeoff between complexity and accuracy.

Anjum et al. [11] proposed a signature based intrusion detection approach for wireless ad hoc networks based on the assumption that attack signatures are completely known in an ad hoc network. This approach investigates the ability of various routing protocols to facilitate the intrusion detection procedure. The authors show that reactive ad-hoc routing protocols are less effective than proactive routing protocols in the detection of intrusions even in the absence of mobility.

Chen et al. [12] proposed a distributed intrusion detection approach based on the Dempster-Shafer theory. They exploit the main advantages of this theory and its ability to reflect uncertainty or a lack of complete information and the convenient numerical procedure for fusing together multiple pieces of data.

Watermarking has been used extensively in the research area of information security. More specifically, in the area of intrusion detection Wang et al. [13] proposed a framework for intrusions detection in wired networks where watermarking and tracing of the packets to the attacker's source IP address is activated, only if the IDS subsystem has determined that there is an attack in progress.

Páez et al. [14] proposed a security scheme for Intrusion Detection Systems based on Cooperative Itinerant Agents (CIA). They proposed a new security scheme in order to verify the entities' integrity of an Intrusion Detection System based on mobile cooperative agents using watermarking software techniques. More specifically, they propose the use of fingerprinting software in order to differentiate agents of the same kind and to detect more sophisticated attacks.

Despite the important advantages that watermarking techniques present no application of watermarking techniques in the area of securing ad hoc networks has been proposed. In this paper we use watermarking in combination with Emergent Self Organizing Maps in order to ensure that the exploitation of the information visualization that eSOM provide will not be altered by malicious attackers. In MANETs the response to possible attacks should be quick as the resources constraints make the security issue an even more difficult task, as the impact of a possible intrusion would be even more severe. This means that information visualization can help us in order to

have a direct response in possible intrusions. Each node has the option to select a secure neighbor node and not one that could be a possible subject of an attack in order to forward its information.

3. Proposed Intrusion Detection Model

Malicious nodes in a mobile ad hoc network may target to exploit features of the physical, network or MAC layers. The majority of the security approaches in such networks have focused in the network layer. Little research has been done on the MAC layer security. The role of the MAC layer in wireless ad hoc networks is substantial as it is responsible for maintaining the communication between nodes and the scheduling of the access in a shared radio channel.

The MAC layer is directly affected by almost every intrusion [8], since it is placed in the first layers of the protocol stack. Indeed, the data delivery ratio, or the throughput, may be affected by malicious behavior or misuse of the shared medium (e.g., selfishness) due to increased routing load. The control overhead for each delivered data packet may also increase. Thus, intrusion detection mechanisms that are based on features selected in the MAC layer are faster regarding the detection delays and the response time. Furthermore, these features make the discrimination between normal and abnormal behavior easier.

The architecture of the IDS applied to MANET could be either distributed and cooperative or distributed and hierarchical. The distributed and hierarchical IDSs are based on dividing the mobile ad hoc network in clusters. Although cluster-based IDSs have the advantage of lower detection workload, the procedure of creating clusters and electing cluster heads may cause a great overhead. Moreover, the existence of cluster heads and the obvious possibility of their exploitation by malicious attackers lead to the weakness of fictitious security. Furthermore the distributed hierarchical IDSs are more efficient for ad hoc networks with low mobility. Thus, the cooperative and dynamic nature of MANETs implies that the intrusion detection system should be distributed and cooperative. The lack of central monitoring nodes and the lack of trust between peer nodes of a wireless ad hoc network render a central intrusion detection system impractical.

Each node of the MANET should perform its local intrusion detection using local audit data. When the confirmation of other nodes to detect an attack is necessary, local intrusion detectors should cooperate. Furthermore, this cooperation between local intrusion detectors should be held through secure channels.

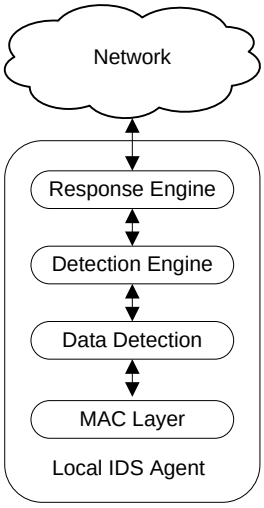

Fig. 1 Intrusion Detection Architecture

The IDS architecture we adopt is composed of multiple local IDS agents as illustrated in Figure 1 that are responsible for detecting possible intrusions locally [4]. The collection of all the independent IDS agents forms the IDS system for the MANET. Each local IDS agent is composed of the following components:

Data Collector: is responsible for selecting local audit data and activity logs.

Detection engine: is responsible for detecting local anomalies using local audit data. The local anomaly detection is performed using the eSOM classification algorithm.

The procedure that is followed in the local detection engine is the one described below:

- Select labeled audit data and perform the appropriate transformations.
- Compute the classifier using training data and the eSOM algorithm.
- Apply the classifier to test local audit data in order to classify it as Normal or Abnormal.
- Perform watermarking in its eSOM map, in order to be sure that it will not be modified and in order to illustrate the security situation and possible existence of intrusions locally in this node.

Additionally, each node selects the eSOM maps of its neighbors and uses them in order to have a view about the security of its neighbors something that can be easily derived by the visual observation of the watermarked (not modified) maps produced by eSOM. After selecting the local maps from its neighbors each node creates the global map of its network consisted of all the local maps and performs watermarking on it. Thus each node is able to know the security status of its local network.

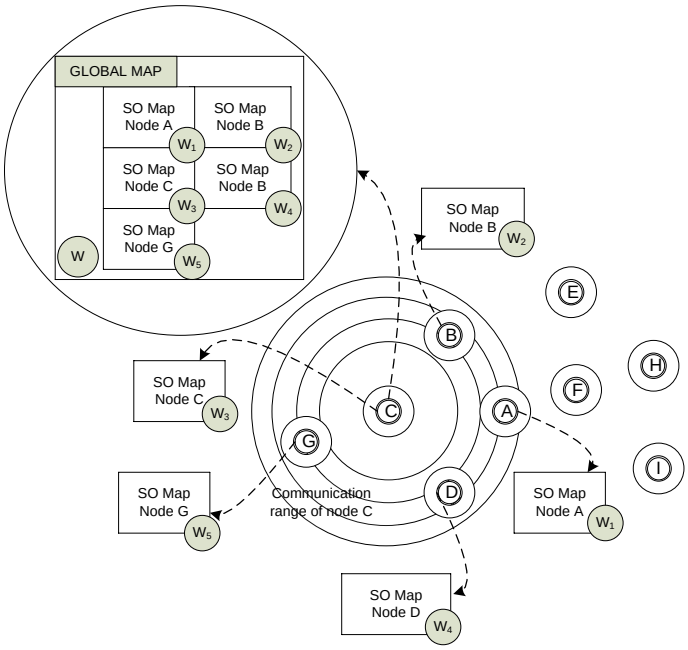

Fig. 2 Watermarked Emergent Self-Organized Maps of a MANET.

In Figure 2, nodes A, B, D and G are in the communication range of node C. Each node A, B, C, D, G creates its own eSOM u-Matrix and performs watermarking on it (illustrated as W_1, W_2, W_3, W_4, W_5 respectively). Node C selects the local watermarked eSOM u-Matrixes from its neighbors and creates the global map of its local network. By observing the global map of its local network, node C is able to have a view of the security status of its neighboring nodes. Based on this information it selects the appropriate node in order to forward its messages. Node C, in order to verify the authenticity and integrity of the global map, performs also watermarking on the global map (illustrated as W). Observing the local maps of all its neighboring nodes and by considering as secure the nodes that are not victims of attacks perform the selection of the appropriate node for the forwarding of messages; their maps do not illustrate the existence of attack. In the case where all nodes are victims of an attack, the node that is considered to be able to forward information is the one that forwards the messages.

Response Engine: If the Detection engine detects an intrusion then the Response engine is activated. The Response Engine is responsible for sending a local and a global alarm in order to notify the nodes of the mobile ad hoc network about the incident of intrusion. Moreover, in case that an intrusion is detected though the local eSOM map of a node then the attacked node is not selected for forwarding information in order to avoid possible loss of information. Special attention should be paid on the function of the Response engine in order to avoid possible flooding caused by the notification messages of intrusion. Thus, the broadcasted notification of intrusion

is restricted to a few hops away from the node where the anomaly has been detected since the neighboring nodes run the greatest risk of possible intrusion.

4. Detection Engine based on Emergent Self-Organizing Maps

Kohonen's Self-Organizing Maps [15] have their base in biology. They belong in the category of unsupervised or competitive learning networks and produce a topological map, which illustrates the input data according to their similarity. The Self Organizing map is trained using only the characteristics of the trained data. The trained KSOMs create clusters of data, where similar vectors of features are located in a specific region in the output space. This is very useful for discovering clusters and relationships in data. The generated mapping is topology preserving.

The learning procedure is composed of the following steps:

a. Initialize the random weights w_{ij} (also known as codebook vectors of the neurons) with small random values.
b. Use an input pattern x.
c. Calculate the Euclidean distance (eq. 1 [15]) between input data sample x, and each neuron weight w_{ij}. The winner (Best Matching Unit) is chosen as $o(x)$:

$$o(x) = \arg\min_j \left\| x - w_{ij} \right\|, \quad j=1,2,\ldots,l, \quad (1)$$

d. Adjust all the weights in the neighborhood, in order to achieve the topological mapping, depending on their distance from the winning neuron according to the following equation [11]:

$$\forall j : w_{ij}(t-1) + a(t)\eta(t') \cdot (x_i(t) - w_{ij}(t-1)) \quad (2),$$

where α is the learning rate, η the neighborhood function and t' the time that was spent in the current context. The neighborhood function η decreases as t' increases.
e. Repeat steps b, c, d until convergence

Something that is often neglected in KSOM is that self-organization allows the emergence of structure in the data. According to [16], "Emergence is the ability of a system to produce a phenomenon on a new, higher level". In order to achieve emergence the existence and cooperation of a great number of elementary processes is necessary. Emergence may be presented not only in natural but also in technical systems. One of the basic disadvantages of SOM maps is that their abilities are limited to a few neurons. On the other hand, emergent Self-Organizing Maps may expand to some thousands neurons. A large number of neurons in eSOM are necessary in order to achieve emergence. The cooperation of such a big number of neurons leads to structures of a higher level. The clustering procedure in emergent SOMs is performed by observing the whole Emergent Self-Organizing Map and not by focusing on its neurons.

We have used the distance based (U-Matrix) method in order to visualize the structures generated by eSOMs. According to this method [16] the sum (height) of distances between the neuron-weights represented as elevation of each neuron. Thus, its

neighbors are normalized by the largest height. The result of the sum of distances is represented as elevation of each neuron. The input data set is displayed and depicted at a 3D landscape. The height will have a large value in areas of the map where few data points belong and small in areas that represent clusters, creating hills and valleys correspondingly. The height ($uh(n_i)$) of each neuron (n_i) is given by the following equation (eq.3) [17]:

$$uh(n_i) = \sum_{n_j \in U_i} d(n_i, n_j) \quad (3),$$

where U_i represent the neighbor neurons of n_i.

We trained Emergent SOMs with logs of network traffic selected from a simulated mobile ad hoc network (using ns-2) and used eSOM U-matrices [17] in order to perform intrusion detection. In our case, a vector represents each log of network traffic with some fixed attributes. Each vector has a unique spatial position in the U-Matrix and the distance between two points is the dissimilarity of two network traffic logs. The U-Matrix of the trained dataset is divided into valleys that represent clusters of normal or attack data and hills that represent borders between clusters. Depending on the position of the best match of an input data point that characterizes a connection this point may belong to a valley (cluster (normal or attack behavior)) or this data point may not be classified if its best match belongs to a hill (boundary). The map that will be created after the training of the Emergent SOM, will represent the network traffic. Thus an input data point may be classified depending on the position of its best match.

Considering the fact that image maps are exposed to the possibility of manipulation, watermarking techniques could be applied to eSOM maps in order to verify the authenticity and detect any modifications of the maps. By using watermarking techniques suspicious parts of images for illegal alterations and modifications could be located.

5. Protecting eSOM maps with Watermarking Techniques

Watermarking techniques attempt to protect the copyrights of any digital medium by embedding a unique message within the original information [1]. The embedding method involves the use of a number of different authentication, encryption and hash algorithms to achieve the integrity and copy protection of the particular message. We use watermarking techniques for eSOM U-Matrixes, which are in the form of images in uncompressed format (Bmp). We use the Lattice and the Block-Wise embedding methods. The Lattice method has two parameters, alpha0 (lattice spacing) and beta (embedding strength) while the Block-Wise method has only one parameter alpha (quantization factor). We combined these two watermarking techniques in order to implement a cryptographic encoder-decoder that can be used in order to authenticate the nodes in the MANET.

One of the most important requirements of watermarking is the perceptual transparency between the original work and the watermarked. In particular for images objective metrics are widely used. The watermark message may have a higher or lower level of perceptibility, meaning that there is a greater or lesser likelihood that a given

observer will perceive the difference between the watermarked and not watermarked image in our case the eSOM u-Matrix.

For fair comparison between the original and the watermarked work there are efficient distortion metrics [3]. Objective criteria are trust worthier in comparison with subjective and they are commonly used in the research and development environment. These distortion metrics do not exploit the properties of the human visual system (HVS) but they provide reliable results. Also there is an objective criterion that relies on the sensitivity of the eye and is called *Watson perceptual distance*. It is also known as just noticeable differences (JND) and consists of a sensitivity function, two masking components based on luminance and contrast masking, and a pooling component. Table 1 gives the metrics that are used more often.

Table 1. Quality Measurements

Mean Square Error	$MSE = \frac{1}{MN} \sum_{m,n} \left(I_{m,n} - \tilde{I}_{m,n}\right)^2$
Signal to Noise Ratio	$SNR = \sum_{m,n} I_{m,n}^2 / \sum_{m,n} \left(I_{m,n} - \tilde{I}_{m,n}\right)^2$
Peak Signal to Noise Ratio	$PSNR = MN \max I_{m,n}^2 / \sum_{m,n} \left(I_{m,n} - \tilde{I}_{m,n}\right)^2$
Image Fidelity	$IF = 1 - \sum_{m,n} \left(I_{m,n} - \tilde{I}_{m,n}\right)^2 / \sum_{m,n} I_{m,n}^2$
Normalized Cross Correlation	$NC = \sum_{m,n} I_{m,n}\tilde{I}_{m,n} / \sum_{m,n} I_{m,n}^2$
Correlation Quality	$CQ = \sum_{m,n} I_{m,n}\tilde{I}_{m,n} / \sum_{m,n} I_{m,n}$
Watson Distance	$D_{wat}(c_0, c_w) = \left(\sum_{i,j,k} \lvert d[i,j,k] \rvert^4 \right)^{1/4}$

The test image is in bitmap format, grayscale and has an 800x600 resolution. In order to use the formulas of Table 1 and to have a view how much is the difference between the original and the watermarked image it was necessary to evaluate the ideal values. Supposing that the original and the watermarked image are exactly identical the produced values are presented in Table 2. The test image is illustrated in Figure 3.

Table 2. Ideal Values of the test Image

Quality Measure-	Ideal
MSE	0
SNR (dB)	94
PSNR (dB)	110
IF	100
NC	1
CQ	138.178
Watson Distance	0

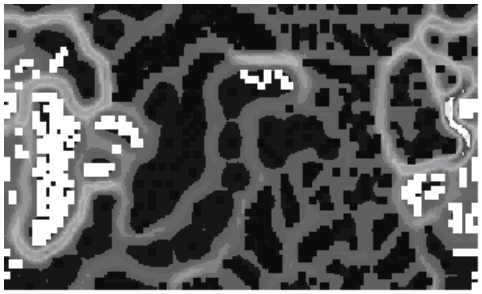

Fig 3. Test Image -Emergent SOM U-Matrix of a node of a MANET

In the following paragraphs the Lattice and the Block-Wise embedding methods are described and how their combination is applied to watermark the eSOM u-Matrixes.

5.1 Lattice Embedding Method

In a lattice code, each code word is a point on a regular lattice. The points in a simple N-dimensional lattice can be constructed by adding integer multiples of N distinct vectors. So each message mark, w_m is a point in a lattice and is the sum of one or more reference marks w_r.

The reference marks are orthogonal to one another. The integer that describes the closest code word to any message vector is calculated, by first finding the length of the message vector projected onto reference mark, and then by diving it by the length and quantizing it to the nearest vector. The lattice watermarking system embeds only one bit per 256 pixels in an image. Each bit is encoded using trellis code and produces sequence of four bits. The trellis coding is a convolutional code, the number of states is 2^3=8 and the possible outputs are 2^4=16. So after the encoding procedure, the bits

have to be embedded in 256 pixels. This means that each of the four bits is embedded in 256/4=64 pixels. The image is divided in blocks of 8x8 pixels in order to host the bits. The reference pattern is constructed consisting of 8x8 random pixels and the pixel values are normalized to have zero mean and unit variance. Each bit is embedded by correlating a block against the 8x8 reference pattern, and quantizing the result to an odd or even integer. The pattern that is added to the 8x8 block according to the index of the closest point in the sublattice ($z_m[i]$) is computed by the following formulas

$$l[i] = \frac{c_i * w_r}{|w_r|} \qquad (4)$$

where c_i is the i^{th} block of the image, w_r is the reference pattern and $l[i]$ is the length of the c_i projected onto w_r

$$z_m[i] = 2\left\lfloor \frac{l[i]/(\beta|w_r|) - m_c[i]}{2} + 0.5 \right\rfloor + m_c[i], \qquad (5)$$

where m_c is the corresponding message

$$w_{ai} = a(\beta z_m[i] w_r - c_i), \quad (6)$$

where $w_{\alpha i}$ is the added pattern.

The parameters in the embedding process are: α (aplha0) that represents the embedding strength and β (beta) that represents the lattice spacing. At the decoder size the $z[i]$ is first computed by equation 7 and then the least significant bit of it is detected. The coded message is then decoded with the trellis decoder.

$$z[i] = \left\lfloor \frac{c_i * w_r}{\beta w_r * w_r} + 0.5 \right\rfloor \quad (7)$$

5.2 Block-Wise Embedding Method

The Block-Wise method involves the basic properties of the JPEG compression where DCT domain takes place. Both the encoder and the decoder use these properties in order to achieve the embedding and the extraction process respectively. The predefined parameters are a strength parameter α (alpha), which is used as the scaling factor of the luminance quantization matrix.

Four bits are embedded in the high-frequency DCT of each 8x8 (64 pixels) block in the image. In the lattice the number of bits, which can be embedded, is one bit per 256 pixels. It seems that using the Block-Wise method the image can host 16 times more information. As it was mentioned the embedding takes place in the high-frequency DCT coefficients and not in the low-frequency in order to avoid any visual differences that would lead to unacceptably poor fidelity. Specifically 28 coefficients are used which means that each bit is embedded in seven coefficients.

The seven coefficients that are going to host one bit are chosen randomly according to a seed number (see Equation 8). So each coefficient is involved in only one bit. Next step is to divide each coefficient by its corresponding quantization factor and round to the nearest integer.

$$C_I[i] = \left\lfloor \frac{C[i]}{aq[i]} + 0.5 \right\rfloor, \qquad (8)$$

where $q[i]$ is the corresponding value of the luminance matrix.

Then the algorithm takes the least significant bit of the resulting seven $C_I[i]$ integers and exclusive-or then to obtain a bit value b_e. The bit value, which has to be embedded, is b. In case that $b_e \neq b$ choose one of the seven integers $C_I[i]$, according which one will cause the least fidelity impact, and flip it. Let $C_{wI}[i]$ denote the result. That is $C_{wI}[i]=C_I[i]$ for all I in case of $b_e=b$ unless $b_e \neq b$, in which case the least significant bit of one member of the seven $C_{wI}[i]$ is multiplied by the corresponding quantization factors to obtain the watermarked versions of the DCT coefficients. The equation is given by

$$C_W[i] = aq[i]C_{WI}[i] \,. \quad (9)$$

At the decoder the procedure is exactly the same. From each 8x8 block the least significant bit b_e is extracted from each of the seven coefficients and is compared with the embedded one b. If they are different, the corresponding block is not authenticated and is marked as corrupted.

5.3 Combined Method

The lattice algorithm uses error control coding and its functionality is based on constructing orthogonal reference marks to be used in the embedding process. But in case that somebody modifies a number of blocks, the decoder will not detect it since it uses trellis coding. Of course if a continuous number of blocks have been changed, the decoder will not be able to extract the correct sequence of bits. The algorithm embeds one bit per 256 pixels. Also the quality of the watermarked image is very high, as it will be discussed in the result sections. On the other hand the block-wise method embeds four bits per 64 pixels. The payload that can be hosted is larger, in comparison with the lattice, and it is very useful in low-resolution images. But the quality of the produced image is not so good. The user can exploit the absence of error control. Any modification of the watermarked image can be located by comparing the extracted message with the original. Questions of whom and why modified the image can be answered easily. So in cases where both the quality and the ability to notice the corrupted blocks have the same importance, it is essentially to combine the two embedding methods.

The combination of the two embedding methods is implemented in a cryptographic encoder-decoder. Then we use a unique feature of the image. This can be anything that characterizes the specific image. As a unique description we used the sum of the pixel values of the four blocks in the corners. These entire three messages are inserted

in a hash function and then the value is encrypted with a 1024-bit secret key. The signature with the short and the extended description are embedded with the lattice method while the message is embedded with the Block-Wise algorithm. The design of the encoder is illustrated in Figure 3.

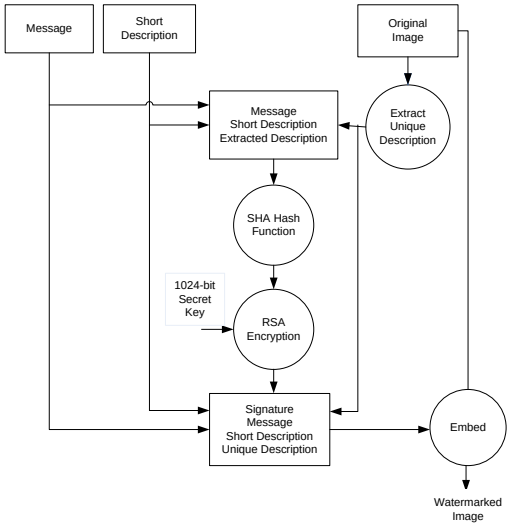

Fig. 3 Cryptographic Encoder

From the watermarked version of the image, at the decoder's side, the signature, the short and the unique description are extracted with the Lattice method while the message is extracted with the Block-Wise method. The unique description is evaluated again but this time of the watermarked version of the work, and is compared with the extracted one. So the first step is to verify if the unique descriptions match. In case of copying the watermark and embedding it in another image, the extracted description will not be the same. Because the pixel values of the image have been slightly changed to host the watermark, the extracted description cannot be exactly the same, but only very close. Therefore some upper and lower boundaries have been determined for this step of verification. The next step is to decrypt the signature using the 1024-bit public key and get the hash value. The message, the short description and the unique description that have been extracted, consist again in the hash function. The obtained hash value is then compared with the one decrypted from the signature. The second step of the decoder is to verify if the decrypted hash value matches exactly with the one calculated at the decoder. If both the stages of the hash values and the unique descriptions are valid, the authentication process is successful. The whole design of the decoder is presented in the Figure 4.

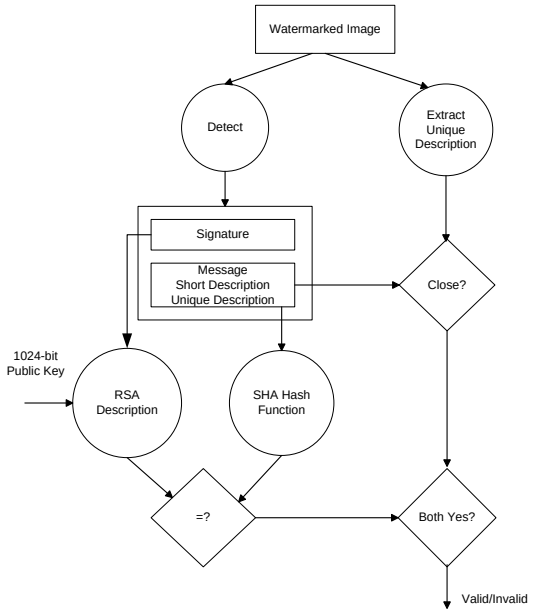

Fig. 4 Cryptographic Decoder

6. Performance Evaluation

6.1 Detection Engine results

To evaluate the feasibility of our intrusion detection engine we have conducted a series of experiments. For our experiments we have made some assumptions. The network has no preexisting infrastructure and the ad hoc routing protocol that was employed is AODV.

We implemented the simulator within the ns-2 library. Our simulation modeled a network of 50 hosts placed randomly within an 1800 × 1000m^2 area. Each node has a radio propagation range of 250 meters and channel capacity was 2 Mb/s. The nodes in the simulation move according to the 'random way point' model. At the start of the simulation, each node waits for a pause time, then randomly selects and moves towards a destination with a speed uniformly lying between zero and the maximum speed. On reaching this destination it pauses again and repeats the above procedure till the end of the simulation. The minimum and maximum speed is set to 0 and 10 m/s, respectively, and pause times at 0, 20, 50, 70 and 200 sec. A pause time of 0 sec cor-

responds to the continuous motion of the node and a pause time of 200 sec corresponds to the time that the node is stationary.

We evaluated the performance of our proposed intrusion detection module for 5, 10, 15 and 20 malicious nodes. In each case the number of all nodes in the network is set to 50. The malicious behavior is carried between 50 and 200 sec. The nodes perform normally between 0 and 50 sec. These parameters result in a network with rather high mobility and high traffic activity.

Twenty, on average, traffic generators were developed to simulate TCP data rate to ten destination nodes. This traffic pattern results in twenty connections among source and destination nodes. The sending packets have random sizes and exponential interarrival times. The sources and the destinations are randomly selected with uniform probabilities. The mean size of the data payload was 512 bytes. Each run is executed for 200 sec of simulation time with a feature-sampling interval of one sec. We used the IEEE 802.11 Distributed Coordination Function (DCF) as the medium access control protocol. The mobility of the nodes is random determined by scenario files that are generated by the scene generator of ns-2. A free space propagation model with a threshold cutoff was used in our experiments. In the radio model, we assumed the ability of a radio to lock onto a sufficiently strong signal in the presence of interfering signals, i.e., radio capture.

In our experiments we have simulated a constant selective packet-dropping attack where the attacker simply discards all data packets while it functions legitimately concerning routing and MAC layer packets. This type of attack is extremely difficult to detect if we consider that packet dropping is due to a malicious behavior or mobility. To add to the problem the malicious node may exhibit malicious behavior when it is most advantageous to him and not from the beginning of the traffic.

The statistical features we have used have been introduced by Liu et al. [10] in their proposed approach for performing intrusion detection in the MAC layer. These features are as follows:

- *Network allocation vector (NAV):* it's a node specific characteristic, which depicts the time that the node will occupy the medium for sending its messages.
- *Transmission traffic rate:* indicates the rate of the transmitted packets.
- *Reception traffic rate:* indicates the rate of the received packets.
- *Retransmission rates of RTS packets:* indicates the rate of the ReadyToSend packets that are retransmitted by the monitoring node. A high value of this feature suggests a possible packet dropping attack.
- *Retransmission rates of DATA packets:* indicates the rate of the data packets that are retransmitted by the monitoring node. A high value of this feature suggests a possible packet dropping attack.
- *Active neighbor node count:* represents the number of neighbor nodes that have data transmission activities.
- *Forwarding node count:* represents the number of neighbor nodes that communicate directly with the monitoring node.

In order to avoid having a great influence of the attributes of some input vectors it is necessary to normalize the input data. Many methods are used for the data normalization. We have normalized the data with mean zero and variance one, a technique

that produces very good results in most cases as reported in the literature. For the evaluation we have used the Databionics eSOM tool ([18], [19]).

The presented evaluation proves that we can achieve a differentiation between normal and abnormal behaviors concerning packet-dropping attacks. In order to perform clustering with eSOM U-Matrices we followed the proceeding procedure. The best matches of the trained dataset and thus the corresponding dataset are manually grouped into clusters representing normal and attack behavior. Thus, we identify the regions of the map that represent a cluster that can be used for the classification on new datasets. The eSOM of a trained dataset is depicted in Figure 3. As it can be clearly seen the training data set has been divided in two classes that are very well distinguished, normal data class (dark color) and packet dropping data class (light color). In order to make sure that our intrusion detection engine will always provide efficient and accurate results we should update our trained eSOM U-matrix according to the new conditions concerning mobility.

In order to evaluate the efficiency of the proposed intrusion detection engine we use two measures the *Detection rate* and the *False alarm rate*:

$Detection\ rate = \frac{TP}{TP + FN}$, $False\ alarm\ rate = \frac{FP}{TN + FP}$, where TP is the number of true positives (attack logs classified as attacks), TN the number of true negatives (normal logs classified as normal), FP the number of false positives (normal logs classified as attacks) and FN the number of false negatives (attack logs classified as normal). The most effective approach should reduce as much as possible the *False alarm rate* and at the same time increase the *Detection rate*.

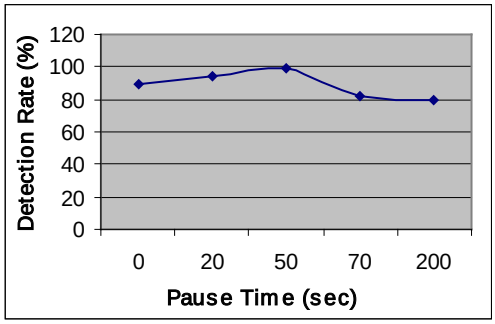

Fig 6. Detection Rate vs. Pause Time

Figure 6 presents the average Detection rate of the all source nodes that present traffic activity and are recognized as normal or attack by eSOM regarding the used pause times. The detection rate seems not to be influenced by the mobility and in all cases to be over 80%. For long pause times the rate slightly lessens which is due to the TCP traffic and the degradation of the mobility. Indeed, a TCP agent stops sending data packets when it doesn't receive acknowledgment. Even after AODV discovers a new path to that destination, the agent keeps sending data packets through the malicious node, as the latter respond normally to control packets. As the network exhibits a

rather low mobility, traffic always is rejected by the malicious node and soon stopped by the TCP agent, which degrades the audit data fed to eSOM.

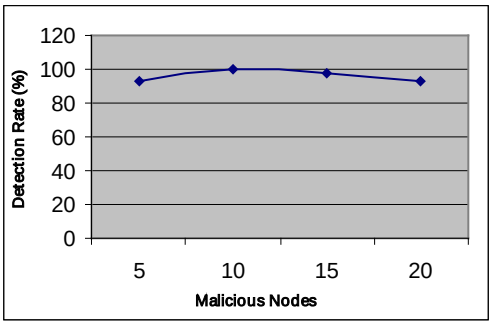

Fig 7. Detection Rate vs. Number of Malicious Nodes

The detection rate as a function of the number of malicious nodes is presented in Figure 7. The rate is rather high and, as in the previous figures, always over 80%. When few malicious nodes exist in the network the connections that are influenced by them are also a few since source nodes move randomly in the network. This results in duplicated lines in the audit data set, which is fed to eSOM thus the decrease in the detection rate. When the number of malicious nodes is high compared to the number of source nodes, the TCP connections generated automatically by NS are a few, which leads to multiple duplicate lines in the audit data that is fed to eSOM, which explains the decrement in the detection rate. TCP traffic is used as a more realistic one. Another data traffic type (e.g. CBR) is under future investigation.

Table 3. False Alarms vs. Pause Time

Pause time (sec)	**False Alarm (%)**
0	21
20	20
50	22
70	20

Table 4. False Alarms vs. Number of Mobile Nodes

Malicious nodes	**False Alarm (%)**
5	26

10	22
15	17
20	21

Table 3 and Table 4 present the average false alarm rate as a function of the paused times used and the number of malicious nodes, respectively. When a source node generates traffic to different destinations and one of these connections is influenced by malicious nodes, then eSOM finds it difficult to distinguish among normal and abnormal traffic. If this is combined with multiple duplicate lines in the audit data due to mobility, the malicious node number produces rather high False alarm rates. The high values of false alarm rates are combated by the activation of the Response Engine, which is able to indicate if the alarm has been triggered by a malicious node or because of mobility issues.

Two representative works in the area of anomaly detection is Deng's et al [6] and Liu's et al [8]. Deng et al [6] in their anomaly detection approach in MANETs they propose the use of SVM (Support vector Machines) in a completely distributed architecture and the false alarm rate range from 3.5±5.8% to 20.85±8.03% for Black hole attack and Frequent False Routing Requesting (FFRR) attack.

Moreover Liu et al [8] in their approach for packet dropping attack using cross feature analysis although the false alarm rate is low 0.29% the detection rate is also rather low 72%. As packet-dropping attack is a rather difficult attack to combat the low false alarm is combined with low detection rate.

Our intrusion detection engine presents rather high detection rate and its main advantage is the visual representation of normal-attack state in a mobile ad hoc network. Moreover our intrusion detection engine has the ability for immediate response in case of possible intrusion by selecting the more secure node as indicated by its u-Matrix map for forwarding the information. In order to verify the reliability and possible alteration of the maps the novel watermarking we propose is used.

6.2 Watermarking Results

In order to evaluate the performance and the efficiency of the embedding methods, excessive tests took place. By understanding the results a clearer idea of the code operation can be acquired as well as a better understanding of the principles behind it. A number of cases will be considered each with a different variable parameter. First the impact of the lattice algorithm on the image quality will be shown. Then the results using the block-wise method are going to be illustrated and finally the section will conclude with the observations using a combination of the embedding methods.

6.2.1 Lattice Embedding Method

In the case of the lattice algorithm the maximum number of the embedded bit can be 400 (one bit per 256 pixels). The formulas that are used to evaluate the differences

between two images were presented in the Table 1. The tests were executed for a range of the parameter's values in order to conclude what the best values are. The parameters are the embedding strength (β) and the lattice spacing (α). The range of the α value was from 0.35 to 5.33 and the range of β from 0.7 to 1.1. The incensement steps for a was 0.02 and for β 0.1. The measurement values for the lattice method are very close to the ideals. More specifically the direction towards zero is achieved using low values of α in case of MSE. If at the same time the value of β that is used is low, the MSE is decreased even further. In the case of SNR and PSNR, the result values are higher when the parameters α and β are low. The image fidelity (IF) is defined as a percentage of how identical the images are. So the value of 100% is considered to be the optimum and as can be noticed from the graphs, the results are very close to this. Utilizing the NC and CQ quality measurements, it is observed that their measurements are closer to ideal ones, as the values of α and β are decreased. Finally all the above observations are also justified from the Watson measurement which is based on luminance, contrast, and pooling masking.

Therefore somebody could suggest that the optimum parameter values are those that give the best results. They could be even the zero values. But at the decoder's side not all the bits are extracted right. Specifically using low values of α and β the decoder is not able to get the right embedded bits. In conclusion it can be said that a trade-off between the quality results and the decoder's result is necessary in order to determine the optimum values. From the tests we concluded that suggested values could be α≈0.8 and β=0.9. In Table 5 are given some evaluated values of the experiments in order to justify all the above notices. The watermarked version of the test image presents no noticeable difference from the test image.

Table 5. Result of the Lattice Method

Lattice	**MSE**	**SNR**	**PSNR**	**IF**	**NC**	**CQ**	**Watson**	**Right Bits**
α=0.35, β=1.0	0.019	64.49	70.21	100	1	137.04	8.144	370
α=1.01, β=0.9	0.27	51.84	56.72	99.996	1	136.97	21.178	400
α=1.85, β=0.8	0.97	49.97	53.12	99.993	1	136.97	51.687	400

6.2.2 Block-Wise Embedding Method

In the case of the Block-Wise method, the tests were executed for the same image in order to be comparable with those for the Lattice method. One major difference is the number of bits that are embedded. Since the method embeds four bits in every 64 pixels and the image has 102500 pixels in total, the number of bits can be hosted in 6406. The size of the information that can be watermarked is significantly higher and in fact is 16 times more than the size for Lattice method. So before even executing the

test it is expected that the results will not be as good. The information is in this case much more, which means that the alterations in the image will produce worse values of the quality measurements. The only parameter in the Block-Wise methods is that which is responsible for the quantization of the luminance matrix and is called alpha (α).

The observation of the results proves what is being stated at the beginning. The values of the quality measurements are not as good in comparison with those from the lattice method. The measurement of the MSE is higher than the zero value, which is ideal. The values of the SNR and PSNR, which are used widely, show that as the value of the parameter alpha (α) is increased the result becomes worse. In the case of the IF, NC, CQ, the measurements seem to be distant from the ideal values as alpha takes higher values. The same conclusion can be phrased for the perceptual distance given from the Watson model, where the results are worse as the value of alpha (α) is increased. Some values of the quality measurements are given in Table 6.

Table 6. Results of the Block-Wise Method

Block-Wise	**MSE**	**SNR**	**PSNR**	**IF**	**NC**	**CQ**	**Watson**	**Right bits**
α=0.03	0.312	44.11	62.18	99.9981	0.99997	138.9	12.144	6012
α=0.16	4.324	36.22	52.32	99.9701	0.99988	137.902	108.972	6406
α=0.33	11.321	31.45	44.29	99.8926	0.99978	137.123	309.456	6406

According to the above paragraph it seems that as the value of alpha is increased, the watermarked image has poorer fidelity. So the optimum value of the parameter could be possibly a small one e.g. 0.01. But it seems that values below 0.01. But it seems that values below 0.05 do no allow the decoder to get the right message. The chosen value of alpha depends on how sensitive the method the user wants it to be in order to locate the corrupted bits and mark the corresponding blocks. Higher values increase the sensitivity but at the same time the quality of the image is reduced. So it is again necessary to make a trade-off between the results and the sensitivity. A possible suggested value could be $\alpha \approx 0.2$. The watermarked version of the original image produced with the Block-Wise method has no visible difference from the test image, which is illustrated in Figure 3.

6.2.3 Combined Method

In order to perform watermarking in the eSOM u-Matrixes we exploit the advantages of the two embedding methods, the Lattice and the Block-Wise method. The Lattice algorithm provides high quality of the watermarked image but the numbers of bits that is embedded in only one bit per 256 pixels. On the other hand, the Block-Wise method embeds four bits per 64 pixels but with the cost of poor quality of the produced image. Furthermore, the absence of error control in the Block-Wise method

gives us the advantage of being able to easily locate any alterations of the watermarked image.

In eSOM u-Matrixes the part that is likely to be illegally altered is watermarked with the Block-Wise method while the rest of the image is watermarked with the lattice method. This means that the areas in the eSOM u-Matrix that are illustrated in Figure 5 with the light color and represent the attack data class in our case the packet dropping data class will be watermarked with the Block-Wise method while the rest of the eSOM u-Matrix (the normal data class (dark color)) with the lattice method. This way, we are able to have a high quality image and at the same time if an adversary changes for example the area of attack data class the combined algorithm is able to determine the modified pixels. This is achieved by comparing the extracted message with the original.

The message is embedded in the part of the image that is watermarked with the block-wise method, while the signature, the short and the extracted description in the large part of the image. The experimental results of the quality measurements were executed again in order to investigate the results. Since the lattice method gives better results than the block-wise, it is expected that the produced result values would be in between the values of those produced by the two methods. Indeed the results were not as good as those of the Lattice's but at the same time better than those of the Block-Wise's. In Table 7 some results of the combination are given in order to compare them with those of the two methods when they are used individually. The table justifies that the combination produces quality measurements between the two methods.

Table 7. Result values of the Combined Embedding Methods

alpha0=0.93, beta=1.0, alpha=0.1	**Lattice alpha0, beta**	**Block-Wise alpha**	**Combined alpha0,beta,alpha**
MSE	0.385	1.785	0.394
SNR	44.2	40.45	45.74
PSNR	53.14	47.25	51.98
IF	99.9972	99.9978	99.9975
NC	0.99999	0.99902	0.99998
CQ	139.457	139.578	139.457
Watson-Distance	31.415	59.788	31.499

alpha0=1.53, beta=0.8, alpha=0.2	**Lattice alpha0, beta**	**Block-Wise alpha**	**Combined alpha0,beta,alpha**
MSE	0.557	4.121	0.74
SNR	44.08	32.97	42.41

PSNR	51.14	40.54	49.75
IF	99.9968	99.9482	99.9836
NC	0.99998	0.99989	0.99997
CQ	139.784	139.78	139.785
Watson-Distance	49.145	155.518	50.002

In Table 8 are presented the maximum number of bits that can be hosted in the image using the two embedding methods and a combination of them.

Table 8. Maximum Number of Embedded bits

	Lattice	**Block-Wise**	**Combined**
Max Embedded Bits	400	6406	>= 4100

We performed a final test in order to verify that in case somebody modifies the block-wised part of the eSOM uMatrix illustrated in Figure 3, the decoder realizes the modification, informs the user that the authentication application failed and outputs a file with the modified blocks marked. The part that is likely to be illegally altered is the light area, which illustrates the attack class. In the watermarked version the light area representing the existence of attack in a node of the MANET was changed and this image was inserted to the decoder in order to verify its authenticity. The authentication process failed and a marked image was produced (Figure 8). By observing the last it is clear that the decoder has successfully located the modified blocks. Therefore the whole implementation of the cryptographic encoder-decoder was correct.

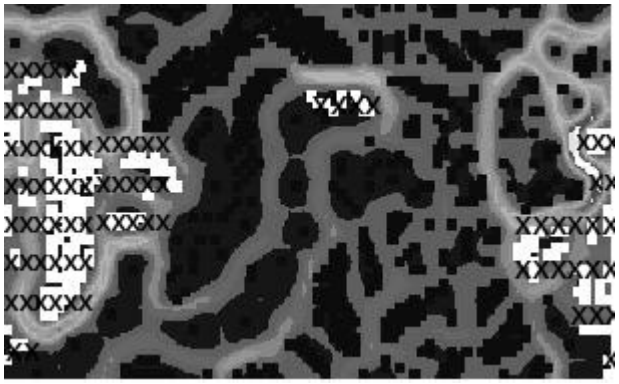

Fig. 8. Marked Image for the Test of the Cryptographic Encoder-Decoder

7. Conclusions and future work

In this paper, we have presented an intrusion detection engine that is part of a local IDS agent that exist in every node of a MANET. The collaboration of all the local IDS agents compose an IDS for MANETs. The proposed intrusion detection engine is based on emergent SOMs a special and efficient class of neural networks that generates as an output a map and provides visual representation of the classification performed. We have examined how eSOM performs in classifying normal and abnormal behavior in MANET based on MAC layer features and we exploited the advantage of visualizing network traffic. We should note that during the classification procedure used in intrusion detection, the classes of the trained data have to be defined manually through the observation of the map something that may introduce a process error.

Using eSOM each node of the MANET creates its local eSOM map as well as the global map of its local MANET. The local and global eSOM maps provide us the important advantage of being able to have a visual representation of the security status of each MANET node as well as its local MANET. Thus, each node has the option to select a secure routing path for packet forwarding by avoiding compromised neighbors.

It is important to note that for the authentication of the local as well as the global maps an innovative and efficient watermarking method is proposed which derives from the combination of two watermarking embedding methods, the Lattice and the Block-Wise. The combined and proposed watermarking method exploits the advantages of the Lattice and the Block-Wise method in order to produce the most efficient and reliable results. The most sensitive part of the eSOM map that represents the existence of an attack in a node being the most sensitive part of the map is watermarked with the block-wise method and the rest of the map with the lattice embedding method.

We exploit the significant advantages of visual representation and watermarking in MANET, two research areas that have not previously used in the research field of MANET. Special attention should be paid to the fact that the detection engine could be employed to various routing protocols. Regarding possible extension of this work, we plan to select features from other layers (e.g. network layer) in order to examine the performance of the proposed approach for the detection of other type of attacks.

REFERENCES

[1] J. Seitz, Digital Watermarking for Digital Media, Information Science Publishing, ISBN: 1591405181, 2005.

[2] Q. Zhang, New techniques for Digital Watermarking, ProQuest / UMI, ISBN: 0542283778, 2006-12-13.

[3] T. Furon, A survey of Watermarking Security, Digital Watermarking: 4h International Workshop (IWDW), LNCS Proceedings, Italy, (2005), pp. 201-215.

[4] Y. Zhang, W. Lee, Y. Huang, "Intrusion Detection Techniques for Mobile Wireless Networks", Wireless Networks 9 (2003), pp. 545-556.

[5] Y. Huang, and W. Lee, "A Cooperative Intrusion Detection System for Ad Hoc Networks", in Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks, October (2003), pp. 135-147.

[6] H. Deng, Q. Zeng, and D. P. Agrawal, "SVM-based Intrusion Detection System for Wireless Ad Hoc Networks", In Proceedings of the IEEE Vehicular Technology Conference (VTC'03), 3 (2003), pp. 2147-2151.

[7] O. Kachirski, and R. Guha, "Intrusion Detection Using Mobile agents in wireless Ad hoc Networks", in Proceedings of the IEEE workshop on Knowledge Media Networking, (2002) pp.153-158.

[8] Y. Liu, Y. Li, H. Man, "MAC Layer Anomaly Detection in Ad Hoc Networks", In Proceedings of 6th IEEE Information Assurance Workshop, June 17, 2005.

[9] Y. Huang, W. Fan, W. Lee, P.Yu, "Cross-Feature analysis for Detecting Ad-Hoc Routing Anomalies", In Proceedings of the 23rd International Conference on Distributed Computing Systems, (2003) pp. 478.

[10] C.Y. Tseng, P. Balasubramanyan, R. Limprasittiporn, J. Rowe, and K. Levitt, "A specification-based Intrusion Detection system for AODV", In Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks, (2003) pp. 125-134.

[11] F. Anjum, D. Subhadrabandhu, S. Sarkar, "Signature-based Intrusion Detection for Wireless Ad-Hose Networks", In Proceedings of Vehicular Technology Conference, Wireless Security Symposium, Orlando, Florida, (2003)

[12] T. M. Chen, V. Venkataramanan, "Dempster-Shafer Theory for Intrusion Detection in Ad Hoc Networks", IEEE Internet Computing, Vol. 9, Issue 6 (November 2005), pp. 35-41.

[13] X. Wang, D.S. Reeves, S.F. Wu, J. Yuill, "Sleepy watermark tracing: an active network-based intrusion response framework", In Proceedings of the 16th International Conference of Information Security (IFIP/SEC'01), Paris, France.

[14] R. Páez, C. Satizábal, J. Forné, "Cooperative Itinerant Agents (CIA): Security Scheme for Intrusion Detection Systems", In Proceedings of the International Conference on Internet Surveillance & Protection (ICISP '06), (2006) p. 26.

[15] S. Haykin, “Neural Networks: A comprehensive Foundation”, Prentice-Hall, New Jersey, USA, 2nd edition (1999).

[16] A. Ultsch, “Data Mining and Knowledge Discovery with Emergent SOFMs for Multivariate Time Series”, In Kohonen Maps, (1999) pp. 33-46.

[17] A. Ultsch, “Maps for visualization of high-dimensional Data Spaces”, Proc. WSOM, Kyushu, Japan, (2003) pp. 225-230.

[18] A. Ultsch, F. Moerchen “ESOM-Maps: tools for clustering, visualization, and classification with Emergent SOM”, Tech. Report Dept. of Mathematics and Computer Science, University of Marburg, Germany, (46) (2005).

[19] Databionic ESOM Tools, Available from <http://databionic-esom.sourceforge.net/devel.html>