Multi-Effector Action Optimized Reinforcement Learning

Mr. S. Senthil Kumar[1] Dr. T. N. Ravi[2]
[1]Research Scholar & Head of Dept. [2]Assistant Professor
[1,2]Department of Computer Application
[1]SCAS, Perambalur, Tamilnadu India [2]Periyar EVR College, Trichy, Tamilnadu, India

Abstract— *Reinforcement Learning (RL) is one of the best Machine Learning (ML) procedures with higher flexibility. RL with Markov's Decision Process (MDP) is performing well in providing automated security with reasonable computational resource consumption. Introducing Multi-Effector Action optimization to Reinforcement Learning reduces standard computer network delays and its positive impact is solid in improving Quality of Service (QoS) of the network. Natural Habitual Learning is an important qualification of Intruder Detection Systems (IDS) in Automated Network Security Mechanisms. Proposed Multi-Effector Action based Reinforcement Learning (MEA-RL) follows more natural habitual Learning. Multiple Prediction and classification processes are swiftly handled in parallel by MEA-RL with lesser power.*

Keywords— Machine Learning (ML), Reinforcement Learning (RL), Markov's Decision Process (MDP), State-Action-Reward-State-Action (SARSA), Multi-Effector Action optimized Reinforcement Learning (MEA-RL)

I. INTRODUCTION

Electronics devices are applied everywhere in today's modern world. Many of these electronic devices are communicating with each other using multiple types of networks. Providing a generalized security for all these devices without affecting the Quality of Service (QoS) is a challenging task. The nodes of the modern network range from high performance network servers to tiny single data transmitting wireless sensor devices [1]. There are many eccentrics of attacks are targeting these devices to steal someone's precious data or to intrude and gain control of a particular network for illegal purposes. This is a real threat to the modern society which relies on computer network totally.

Thousands of new devices are included to the existing networks day-to-day. All these devices are of different types and equipped for different purposes. Having heterogeneous types of network infrastructure is common these days. These nodes can be classified into different types based on their power and bandwidth utilization. Network Servers, terminal computers, multi-functional devices, mobile phones, wireless sensor devices and Internet of Things are some of the major classifications of latest network nodes. Every classification has their own security policy using different types of cryptography, protocols and different communication standards. Interconnecting all these devices through a same network is a frequent requirement to achieve higher flexibility and usage. Providing security without disturbing the QoS is a complicated task. Monitoring present time networks using network-monitoring tools and providing real-time security manually is merely an impossible task. Therefore scientist and developers are involved in devising automated security solutions for decades. The premium qualities of these automated security systems are improving existing security as well as improving the network quality. The quality of a network is determined by a number of parameters like throughput, latency, jitter, end-to-end delay, power consumption and security level. These parameters can be measured using a Network Simulator like OPNET[2].

II. RELATED WORKS

Providing dynamic security for heterogeneous network can be modelled in deterministic and non-deterministic categories. As well the classification process can be automated using data mining techniques or neural network designs. Both the methods require a huge dataset to train the system to get highest accuracy levels. Improved Artificial intelligence [3] [4] procedures facilitate to train automation system with less number of sample data. Major security automations are done using Active Reinforcement Learning (ARL), Reinforcement Learning (RL) and Reinforcement learning with Markov's Decision Process (RLMDP).

A. Active Reinforcement Learning (ARL)

Active Reinforcement Learning is based on direct utility estimation and adaptive dynamic programming. Bellman's equation

$$U^{\pi}(s) = R(s) + \gamma \sum_{s'} T(s, \pi(s), s') U^{\pi}(S') \quad (1)$$

is used to validate the utility estimations, where U is the hypothesis space and $T(s, \pi(s), s')$ is estimated through trials.

The Temporal Difference (TD) is initialized with average probabilistic transition as the following equation

$$U^{\pi}(s) \rightarrow (1-\alpha)U^{\pi}(s) + \alpha\left(R(s) + \gamma U^{\pi}(s')\right) \quad (2)$$

Learning rate α determines the convergence of utility. Convergence is identified using the following conditions

$$\sum_{m=1}^{\infty} \alpha_s^1(m) = \infty \quad \sum_{m=1}^{\infty} \alpha_s^2(m) < \infty \quad (3)$$

Final Active Reinforcement Learning optimal utility action is calculated using

$$U(s) = R(s) + v_a^{max} \sum_{s'} T(s, a, s') U(s') \quad (4)$$

ARL follows filtered active approaches solely, this makes it simple to use in many applications and they adopt average learning capabilities.

B. Reinforcement Learning (RL)

Reinforcement Learning overcomes the disadvantages of ARL and performed well with dynamic independent data. RL combines both active and passive approach learning simultaneously. RL replicates the real learning nature of midbrain dopamine that learns by performing reward oriented prediction. Each knowledgebase entry of RL resembles the actual firing of a dopamine neuron. Periodical updates based on 'State-Action and Reward-State-Action' are performed in knowledgebase of RL. This arrangement makes it possible to train RL more efficiently even with independent data. This machine learning procedure found to

be fittest Artificial Intelligence (AI) method to develop network security.

RL considers all 41 essential network security data metrics. They are duration: continuous, protocol_type: symbolic, service: symbolic, flag: symbolic, src_bytes: continuous, dst_bytes: continuous, land: symbolic, wrong_fragment: continuous, urgent: continuous, hot: continuous, num_failed_logins: continuous, logged_in: symbolic, num_compromised: continuous, root_shell: continuous, su_attempted: continuous, num_root: continuous, num_file_creations: continuous, num_shells: continuous, num_access_files: continuous, num_outbound_cmds: continuous, is_host_login: symbolic, is_guest_login: symbolic, count: continuous, srv_count: continuous, serror_rate: continuous, srv_serror_rate: continuous, rerror_rate: continuous, srv_rerror_rate: continuous, same_srv_rate: continuous, diff_srv_rate: continuous, srv_diff_host_rate: continuous, dst_host_count: continuous, dst_host_srv_count: continuous, dst_host_same_srv_rate: continuous, dst_host_diff_srv_rate: continuous, dst_host_ same _src_port_rate: continuous, dst_host_srv_diff_ host_rate: continuous, dst_host_serror_rate: continuous, dst_host_srv_serror_rate: continuous, dst_host_rerror_rate: continuous, dst_host_srv_rerror_rate: continuous.

All these parameters are involved in calculating decision making factors for RL. Expected sum of immediate and long-time rewards under the more suitable policy referred as Utility. It is calculated as

$$\text{util}(s_t, a) = E\left\{\text{Reward}(s_t, a) + \max \text{policies} \sum_{j=1}^{N-1} R_{t+j}\right\} \quad (5)$$

Where s_t refers the state at particular timestamp t, $\text{Reward}(s_t, a)$ refers immediate reward of executing action a in state s_t, N refers number of steps taken by the agent in its lifetime. E{.} refers expectation over all possible combination of decisions.

Sometimes RL abides by taking reward oriented heuristic decisions makes the security system vulnerable to strategic long term attacks. In this criterion RL needs larger time consuming updates in its knowledgebase which makes the security system less responsible to the real-time data.

C. Reinforcement Learning with Markov's decision Process (RLMDP)

RL's time consuming knowledge base updates against 'strategic attacks' are optimized in RLMDP. Markov's Decision Process also reduces the number of heuristics movements of RL. Whenever there is an ambiguous decision or a decision with less support count occurs, RL took the default action expecting a reward whereas RLMDP applies MDP and filters the action if there is a less probability to get the reward. This nature of RLMDP makes it more stable against different attacks.

Markov Decision Processes (MDPs)[6][7] are operates on high dimensional state and action spaces represented as s and a respectively. To get the state s_t, action a_t and reward r_t at time t the state transition combinational probabilities and expected reward function is declared as $P(s_{t+1}|s_t, a_t)$ and $R(s_t, a_t)$. Stochastic and stationary policies declared by conditional distributions over actions $\pi^\theta(a; s)$ parameterized by θ. It is assigned that given policy π^θ the MDP is ergodic with stationary distribution d^θ. In RLMDP energy-based policies are considered which can be expressed as conditional joint distributions over actions a and a set of latent variables h

$$\pi^\theta(a, n; s) = \frac{1}{Z(s)} e\phi(s, a, h)\top_\theta \quad (6)$$

Where φ(s,a,h) are a pre-defined set of features and $\Sigma_{a,h}\exp(e\phi(s, a, h)\top_\theta)$ is the normalizing partition function. The policy itself is then obtained by marginalizing out h. Latent type variables used to make policies based on energy and these policies classify composite non-linear and non-product relationship between actions and states inherent classification (1) is log linear with the features φ(s,a,h). In a conditional Restricted Boltzmann Machine (RBM), the states s, actions a and latent variables h are all high dimensional binary vectors, and (1) is parameterized as

$$\pi^\theta(a, n; s) = \frac{1}{Z(s)} e^{s\top w_s h + a^\top w_a h + b_s \top_s + b_h \top_h + b_a \top_a} \quad (7)$$

Where the parameters are matrices Ws, Wa and vectors bs, ba, bh of appropriate dimensionalities. Marginalizing out h, used to get a non-linearly parameterized policy

$$F^\theta(s, a) = -b_s \top_s - b_a \top_a - \Sigma_i \log\left(1 + e^{s^\top w_{si} + a^\top w_{ai} + b_{hi}}\right),$$
$$\pi^\theta(a; s) = \frac{1}{z(s)} e^{-F(s,a;\theta)} \quad (8)$$

Where i indices the latent variables, and w_{si}, w_{ai}, b_{hi} are parameters associated with latent variable h_i. The quantity $F(s, a; \theta)$ is the conserved energy.

The policy selection is constantly updated by SARSA, the state action pairs can be the nearest neighbor nodes. Here physical position of cluster information is used instead of Virtual Power Cluster (VPC) to reduce computational power. The error rate of SARSA can be computed as

$$\varepsilon(s^t, a^t) = [r^t + \gamma Q(s^{t+1}, t^{t+1})] - Q(s^t, a^t) \quad (9)$$

In case the state-action function is determined by θ, then the update equation for new parameter is

$$\Delta\theta \propto \varepsilon(s^t, a^t)\nabla_\theta Q(s^t, a^t) \quad (10)$$

The update process determines the security policy $M_x(R)$for the corresponding cluster P_x. The RL system was pre-trained to a beginning level with the optimal action

$$P(a|s) \approx \frac{e^{Q(s,a)/\tau}}{z} \quad (11)$$

Where z is a normalization factor, τ is a positive number represents the iteration. The RL convergence can be identified with the value of τ, if it is getting higher values, then it refers the RL training is under progress is with uniform improvement.

Above the all existing methods, RLMDP is the only procedure that operates with power awareness. Many of the present network nodes are battery operated. Therefore providing security along with less power consumption is also important. The concept of Virtual Power Clusters (VPC) [8] [9] is used in RLMDP to facilitate a balanced action between power and security. The lack of parallelism and linear State Action – Reward State Action are main disadvantages of RLMDP and this affects the performance of RLMDP while dealing real-time dynamic data.

III. PROPOSED METHOD & IMPLEMENTATION

Multi-Effector Action Reinforcement Learning (MEA-RL) is designed to perform State Action and Reward State

Action [10] [11] in parallel. MEA-RL is running in all cluster heads. A network transaction is referred by all 41 benchmark connection request parameters. MEA-RL maintains a sandboxed environment with regular updates. Whenever an ambiguous network transaction arrives, MEA-RL performs heuristics action of blocking the transaction and permitting it in the sandboxed environment [Figure 1].

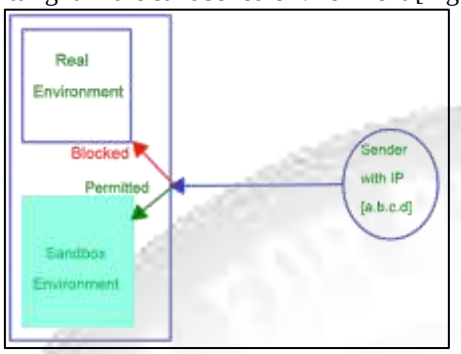

Fig. 1: Heuristic action of blocking the transaction

It monitors the activities of the particular established virtual connection for a while and if there is any malicious activity found [Figure 2], it restores the sandboxed environment and left the connection blocked. Knowledge about the particular transaction is added to the knowledgebase.

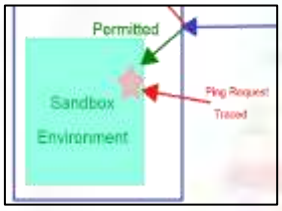

Fig. 2: Malicious Activity

While monitoring the sandboxed environment after performing the virtual connection, if there are no malicious activities [Figure 3], then the MEA-RL generates a resend request to the suspected sender and permits the connection if the transaction details are replica of previous one [Figure 4]. It also adds the sender details to the trusted list tables.

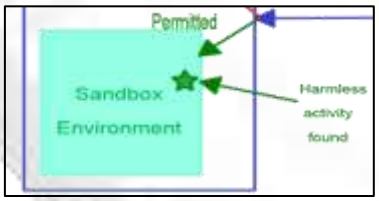

Fig. 3: No Malicious Activities

In MEA-RL, if $P(s)$ is a decision making policy with any of the mapping from states to actions, then the policy action quotient Q^P can be calculated as

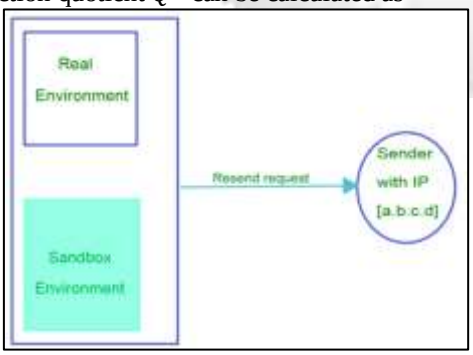

Fig. 4: Resend Request

$$Q^P(s_t, a_t) = E[r1+\gamma r_{i+1} + \gamma^2 r_{i+2} + \gamma^3 r_{i+3} \ldots | s_i, a_{i,}, p]m \quad (12)$$

Futures states can be calculated by performing recursive form as

$$Q^P(s_t, a_t) = r(s_t) + r \sum_{s_t+1 \in \varphi} P(s_{t+1}|s_t, a_t)\, Q^P(s_{t+1}, P(s_{t+1})) \quad (13)$$

Optimal value function along with associated policy can be calculated as

$$Q^*(s_t, a_t) = r(s_t) + r \sum_{s_t+1 \in \varphi} P(s_{t+1}|s_t, a_t) \max_{a_{t+1} \in A} Q^*(s_{t+1}, a_{t+1}) \quad (14)$$

Consider the mean-estimate rule is $\mu_k(s_k)$, then error driven mean-estimation is calculated using

$$\mu_{k+1}(s_k) = \mu_k(s_k) + k_k \cdot \delta_k \quad (15)$$

Where k_k is knowledge base update rate or it can be referred as the learning rate also. k_k is calculated using

$$k_k = \frac{\sigma_k^2(s_k)}{\sigma_k^2(s_k) + \sigma_r^2(s_k)} \quad (16)$$

Prediction error is calculated as $\delta_k = \gamma_k - \mu_k$

IV. PERFORMANCE ANALYSIS

Performance of MEA-RL along with existing methods are measured by calculating the standard network QoS parameters throughput, latency, jitter, end-to-end delay, security level and average power consumption. Ten equal time stamps are selected from the simulation process. Active Reinforcement Learning, Reinforcement Learning, and Reinforcement Learning with MDP are taken as the participants in the simulation to compare with the proposed Multi-Effectors Action Reinforcement Learning. An User Interface and script wrapper to OPNET Modeler[12][13] is designed with Visual C++ programming language of Visual Studio 2013 Integrated Development Environment(IDE). NETSIMCAP – a Network Simulation and Capture Software Development Kit is used to interface Visual C++ with OPNET network simulator. Centralized Server with six Wi-Fi routers and 50 heterogeneous nodes are placed using random placement scheme [14] of OPNET to create a hybrid heterogeneous network[15][16] environment.

Figure 5 shows the placement of heterogeneous nodes and wireless network distribution hotspots in OPNET.

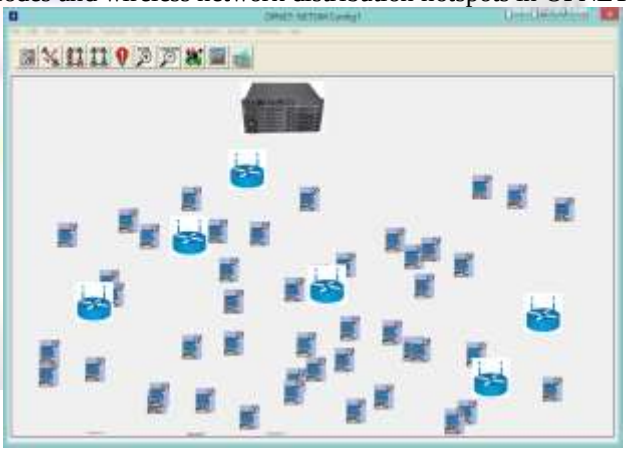

Fig. 5: OPNET Network structure

Throughput of ARL, RL, RLMDP and MEA-RL are shown in Figure 6. ARL achieved throughput from 800 Mbps to 990 Mbps based on the random locations of the nodes. Reinforcement Learning achieved a little better than ARL, i.e. from 1100 Mbps to 1215 Mbps. RLMDP got the throughput range of 1200 Mbps to 1350 Mbps which is

higher than ARL and RL. Whereas proposed MEA-RL achieved the highest throughput of 1450 to 1550 Mbps which is higher than all other methods taken into comparison that is shown in Figure 6.

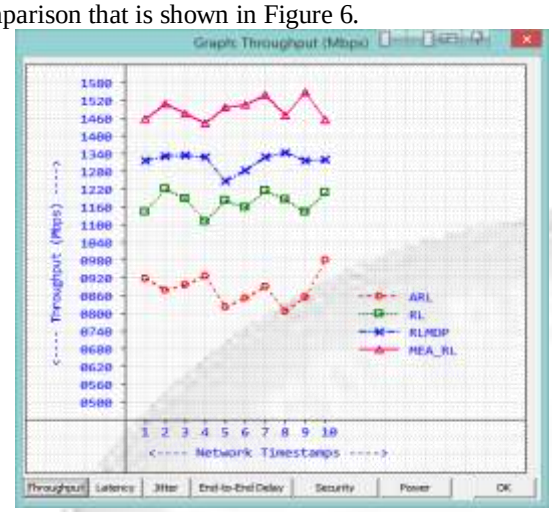

Fig. 6: Throughput (Mbps)

Latency is a delay in nodes response for a network transaction. This latency is inversely proportional to QoS of a network. To maintain a better QoS, latency has to be kept to the bare minimum negligible level. MEA-RL reduces the latency to the minimum value of 133 mS whereas other existing methods took 150 to 250 mS. Latency comparison chart of existing and proposed methods is given in figure 7.

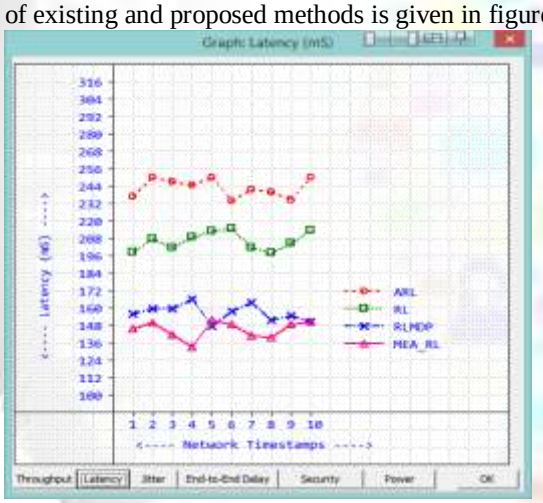

Fig. 7: Latency (mS)

The time difference in packet inter-arrival time to their destination is called as jitter. Jitter is a natural delay in packet based network communication. In general, TCP and IP protocols are dealing with the jitter impact on communication. To achieve higher QoS, jitter should be kept to the minimum negligible level. The lowest value of 32mS is achieved by the MEA-RL implies the higher performance than the other methods involved. Comparison graph shown in Figure 8.

End-to-end Delay is the average travelling time taken by a data packet from source to destination. It includes delays caused by route discovery process and the data packet transmission queue. Dropped packets are not considered while calculating end-to-end delay and all successfully delivered packets are included in the end-to-end delay calculation. The measured End-to-End delay of MEA-RL method is shown in Figure 9. MEA-RL gets the minimum end-to-end delay of the range from 498mS to 532mS.

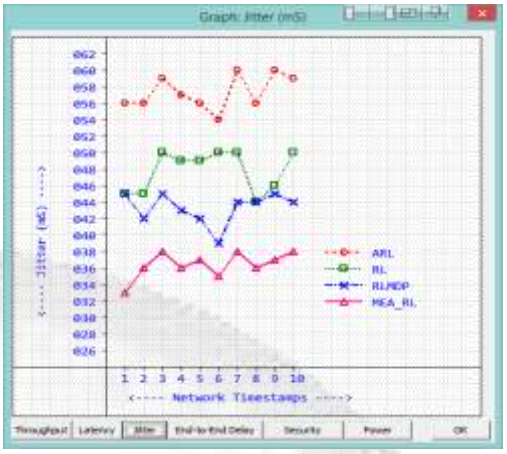

Fig. 8: Jitter (mS)

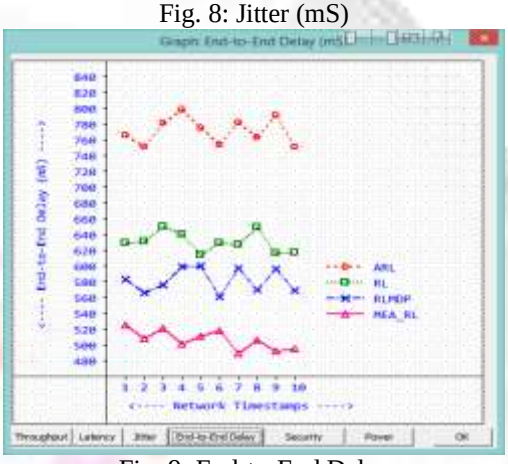

Fig. 9: End-to-End Delay

Security is the vital criteria involved in modern networks with shared and distributed infrastructures. The higher security level refers the higher quality of the network architecture. The highest security value of 96% is achieved by MEA-RL is shown in Figure 10. Even though RL and RLMDP are getting closer security levels with the security level of proposed Multi-Effector Action Based Reinforcement Learning (MEA-RL), higher category average is achieved by MEA-RL.

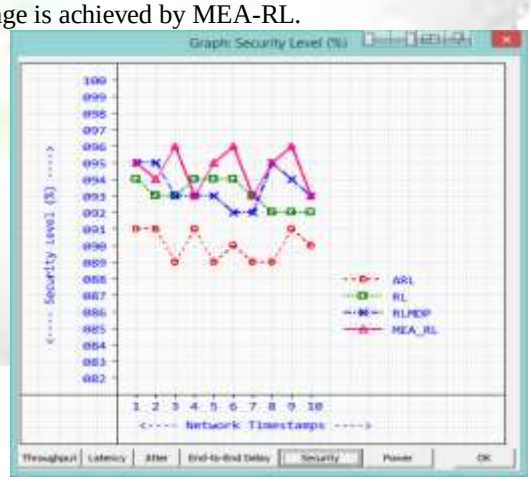

Fig. 10: Security Level (%)

The prime target of proposed method is to provide uncompromising Quality of Service (QoS) with highest security and lowest power consumption. The proposed Multi-Effector Action optimized Reinforcement Learning (MEA-RL) achieved the prime target. Based on the OPNET

simulation MEA-RL is measured with the lesser power consumption range from 576mW to 664mW. Average power consumption of existing method and proposed method are compared in a graph shown in Figure 11.

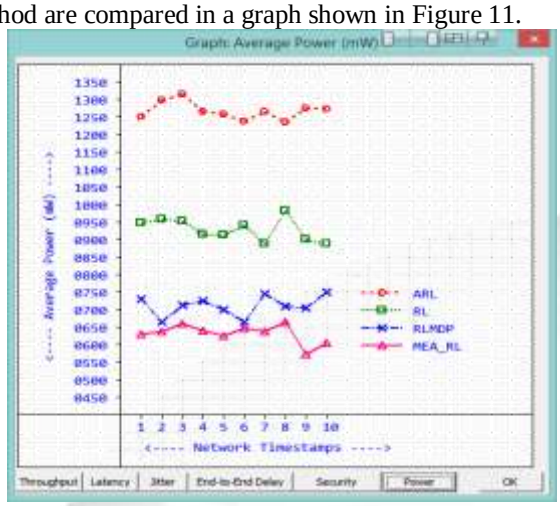

Fig. 11: Average Power (mS)

V. Conclusion and Future Work

In this paper, the Reinforcement Learning method is endued with Multi-Effctor Actions. The proposed method is indented to create a balance between the power and security in modern heterogeneous networks. As per the implementation results, it is clearly observed and stated that the proposed MEA-RL improves the security level, throughput while reducing latency, jitter, end-to-end delay and power consumption. This MEA-RL based automated security system for modern networks will improve the end user experience so as saves huge computational resources and energy than any other automatic machine learning schemes. Dividing the intact network into minor subgroups and introducing sub-centralized servers to create a micro-level knowledge base generation, the security policy selection patterns can be studied clearly by combining the cumulative knowledge base into a single entity. This singularity can be achieved and an optimum hybrid Network can be created for the commercial and other essential purposes.

References

[1] S. Özdemir, "Secure data aggregation in wireless sensor networks via homomorphic encryption,"Journal of the Faculty of Engineering and Architecture of Gazi University, vol. 23, no. 2, pp. 365–373, 2008.

[2] Adarshpal S. Sethi and Vasil Y, "The Practical OPNET User Guide for Computer Network Simulation" Hnatyshin Taylor Francis Ltd: ISBN13: 978143981205 1- ISBN10 : 1439812055

[3] M. M. Hamada, Al-Minia University, Egypt; M. A. A. Wahab, N. G. A. Hemdan, "Artificial Neural Network Modeling Technique for Voltage Stability Assessment of Radial Distribution Systems". Proceedings of the 41st International Universities Power Engineering Conference, (Volume:3).978-186135-342-9. Publisher: IEEE, 2006.

[4] R. Zhang, "Artificial intelligence techniques with modern logistics applications". Department of Basic Subjects, West Anhui Health Vocation College, Lu'an 237000, China, S. Su ; Xie Zhengao, ISBN: 978-1-4244-7616-9, Publisher:IEEE, 2010.

[5] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set", Proposed of the IEEE, 2009

[6] K. V. Katsikopoulos , "Markov decision processes with delays and asynchronous cost collection". Dept. of Mech. & Ind. Eng., Massachusetts Univ., Amherst, MA, USA; S. E. Engelbrecht. ISSN: 0018-9286 Publisher: IEEE, 2003.

[7] T. Campbell, "Multi-agent allocation of Markov decision process tasks". Department of Aeronautics and Astronautics, MIT, USA; L. Johnson; J. P. How. ISBN: 978-1-4799-0177-7. Publisher: IEEE Published in: American Control Conference, 2013.

[8] D. Jin, "Fast Complex Network Clustering Algorithm Using Agents". College of Computer Science & Technology, Jilin Univ., Changchun, China ; D. Liu ; B. Yang; J. Liu. ISBN: 978-0-7695-3929-4. Publisher: IEEE, 2009.

[9] D. Bortner, "Progressive clustering of networks using Structure-Connected Order of Traversal". Department of Computer Science, University of Illinois at Urbana-Champaign, USA; J. Han. Print ISBN: 978-1-4244-5445-7. Publisher: IEEE, 2010.

[10] HH Viet, PH Kyaw,"Simulation-based evaluations of reinforcement learning algorithms for autonomous mobile robot path planning". TC Chung - IT Convergence and Services, Springer, 2011

[11] J Gläscher, N Daw, P Dayan, JP O'Doherty, "States versus rewards: dissociable neural prediction error signals underlying model-based and model-free reinforcement learning". Neuron, Elsevier, 2010.

[12] M. Fazeli ; Imam Hossein Univ., Tehran, Iran ; H. Vaziri. "Assessment of Throughput Performance under OPNET Modeler Simulation Tools in Mobile Ad Hoc Networks (MANETs)". Published in: Computational Intelligence, Communication Systems and Networks (CICSyN), Third International Conference. ISBN: 978-1-4577-0975-3. Publisher: IEEE, 2011.

[13] I. S. Hammoodi, "A Comprehensive Performance Study of OPNET Modeler for ZigBee Wireless Sensor Networks". Caledonian Coll. of Eng., Muscat, Oman ; B. G. Stewart ; A. Kocian ; S. G. McMeekin. ISSN:2161-2889. ISBN: 978-0-7695-3786-3. Publisher:IEEE, 2009.

[14] M. S. Hasan, "Simulation of Distributed Wireless Networked Control Systems over MANET using OPNET". Faculty of Computing, Engineering and Technology, Staffordshire University, UK, m.s.hasan@ staffs.ac.uk; H. Yu; A. Griffiths; T. C. Yang. Print ISBN:1-4244-1075-4. Publisher: IEEE, 2007.

[15] A. Khandekar, "LTE-Advanced: Heterogeneous networks". Qualcomm Inc., 5775 Morehouse Drive, San Diego, CA 92121 U.S.A. ; N. Bhushan ; J. Tingfang ; V. Vanghi. Print ISBN:978-1-4244-5999-5. Publisher:IEEE

[16] E. Bodanese, "A brief introduction to heterogeneous networks (HetNets) and its challenges". School of Electronic Engineering and Computer Science, Queen

Mary, University of London, United Kingdom. Published in: Communication Technology and Application (ICCTA), IET International Conference. Publisher: IET, 2011.