Received: 6 March 2021 | Revised: 8 September 2021 | Accepted: 30 September 2021

DOI: 10.1002/int.22708

RESEARCH ARTICLE

WILEY

Deep reinforcement learning for building honeypots against runtime DoS attack

Selvakumar Veluchamy[1] | **Ruba Soundar Kathavarayan**[2]

[1]Department of Computer Science and Engineering, Chennai Institute of Technology, Chennai, Tamil Nadu, India

[2]Department of Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi, Tamil Nadu, India

Correspondence
Selvakumar Veluchamy, Department of Computer Science and Engineering, Chennai Institute of Technology, Chennai, Tamil Nadu, India.
Email: vselvakumar0720@gmail.com

Abstract

Honeypot is a network environment utilized to protect proper network sources from attacks. Honeypot makes an environment that attracts the attacker to pay their operations to steal sources. Denial of Service (DoS) attacks are efficiently noticed using the proposed honeypot method. The issues of the previous technique are that the DoS attack is a malicious act with the goal of interrupting the access to a computer network. The result of the DoS attack can cause the computers on the network to squander their resources to serve illegitimate requests that result in a disruption of the network's services to legitimate users. To overcome these challenges this method is proposed. In this manuscript, the Deep Adaptive Reinforcement Learning for Honeypots (DARLH) is proposed. Here, honeypot environment, the proposed DARLHs system implements Deep Adaptive Reinforcement Learning (DARL) with Intrusion Detection System (IDS) agents and Deep Recurrent Neural Network (DRNN) with IDS agent for observing multiruntime DoS attack. In the next level, the system creates DRNN and DARL IDS agent integration modules for effective runtime attack detections. The Knowledge Data Discovery data set pattern, UNSW-NB20, and Bot-IoT data sets are used to the scenario of DoS attack. The method is executed in Python 3.7. The experimental outcomes are likened through different existing methods, such as Game and

Naïve-Bayes Honeypot, Block Chain Honeypot, and Recurrent Neural Network-based Signature Generation and Detection. The proposed method is compared with External DoS Attack, Internal DoS attack, Brute-force attack, DoS attack, Web attack, and Botnet attacks with the existing methods. From the comparison, the proposed method offers 5%–10% better outcomes than another existing method. Lastly, the test results determine that the proposed method performance is most efficient with another existing system.

KEYWORDS
deep learning techniques, denial of service, event tracking, honeypot, intrusion detection system agents

1 | INTRODUCTION

Honeypot is a secure machine or a collection of machines to appeal invaders into it. Honeypot systems are the part of enterprise networks, but deployed as independent blocks. These systems are having a portion of data in any machine to attract the attackers to start their hacking activities. Honeypot machines monitor the activities injected by an attacker or any intruder continuously. The honeypot system maintains a sufficient number of hardware and software resources to execute the attack monitoring activities. Also, the system contains the huge active and passive attacker data sets.[1–4]

Particularly, honeypot creates a duplicate and isolated network to validate the external traffics. It helps improve the security of main networks.[5–7] Moreover, it helps implement the well-defined Network Intrusion Detection System (NIDS). This honeypot depends on NIDS agents' implementation can be deployed centrally or distributed manner. Many related research works put their effort into honeypot-based Intrusion Detection System (IDS) agents to detect various attacks. Among these attacks, Denial of Service (DoS) is a serious active network attack to dump the network functions.[8,9] The recent researches are developed IDS agents against DoS using rule-based classification techniques, Support Vector Machines (SVMs), and other Machine Learning (ML) techniques.

Game and Naïve-Bayes Honeypot (GNBH)[10,11] is the susceptibility of the route conservation phase in a wireless mesh network, and the equivalent attack technique is considered for the network protocol. The Bayesian honeypot game model is used to respond to the limitations of the traditional honeypot method detection. Block Chain Honeypot (BCH)[12] provides the solution of the problem affected by centralization, distributed and decentralized manner, which develops block chain that features decentralization. The data in block chain are tamper resistant. RNN-based Signature Generation and Detection (RNSG) is to industrialize protection devices, once required by the generation of online detection as well as zero-day attacks, so valuable data on resources, such as web and email servers, can be preprotected before these attacks. Here, the risk to the environment is introduced, whereas honeypot once attacked, it can be used to attack, infiltrate, or harm other systems or organizations. The techniques with a sufficient level of trained data set provide additional intelligence to honeypot systems. This improves the detection rate of attacks. The honeypot network will affect the problem, such as influence of disturbances, modeling errors, and various uncertainties in the real systems which will decrease the robustness of the system and reduce the security due to various attacks entering into the system. The attacks that are deception attacks, DoS attacks, and false information injection attacks are the frequently happened phenomena done with honeypot networks. These attacks must consider adequately through system analysis/synthesis to prevent involuntary behaviors (e.g., degradation, divergence, and instability). In addition, in practical applications (real systems), several factors can cause uneven changes in communication networks topology, for example, communication link instability among filters, obstacle blocking, network-induced factors, and adding new nodes due to application supplies. In recent years, various attainments have been made on problem.[14–19] Some of the papers are explained below to improve the robustness of the Honeypot network and to enhance the security.

Impulsive reaction-diffusion neural networks (NN) input-to-state stability through infinite dispersed delay utilizing an impulsive stochastic fuzzy class delayed Cohen–Grossberg neural networks (C-GNNs) with distributed infinite transmission delay. Through Razumikh in method as well as difference disparity, some sufficient conditions ensuring the mean-square exponential input-to-state stability of consider C-GNN is attained. These methods provide less security in forensic applications.[13] In Nonlinear Dynamics a robust PD-type of learning control for unique systems through multi delays is subject to poly topic uncertain as well as limited frequency-domain. The multiscale hybrid nanocomposite (MHC) disk (MHCD) is examined inactive on an elastic base subject to nonlinear temperature gradient as well as the mechanical load for nonlinear frequency analysis. This method is limited due to hackers entered into the system and authentication is enabling, allowing a remote attacker to control the user accounts existence. The limitation of this method is that it only secures a limited attack,[14] Multidimensional Systems and Signal processing. In this manuscript, new processor algebra depends on techniques for test the $n-D$ discrete linear systems structural stability (with $n \geq 2$). This method has the limitation due to highly expensive and less security,[15] Fuzzy Fault Detection for Markov Jump Systems through Partly Accessible Hidden Information. This study explores the error detection of asynchronous filtering for homogeneous Markov jump systems through discrete-time piecewise. By resort to a double hidden Markov system, an asynchronous fault detection filter is provided that can follow system methods. This method has a limitation of less security.[16] The event-triggered tracking regulator second-order error multi agent systems in system non-linearity considered is examined through using dispersed sliding-mode control (SMC) approach. The event-triggered tracking regulator second-order error multiagent system in system nonlinearity considered is examined through using dispersed sliding-mode control (SMC) approach. An event-triggered strategy is to reduce the frequency of the controller model and store network communication resources; the trigger state is established with the following multiple agent system for a leader. This method has a limitation of the more attacks involved in the system and this method does not provide security against attacks,[17] Adaptive Optimization Algorithm for Nonlinear Markov Jump Systems through Partial Unknown Dynamics. In this manuscript, an online adaptive optimal control error of a continuous-time class Markov jump linear system (MJLS) is examined by utilizing a parallel reinforcement learning (RL) algorithm by completely unknown dynamics. Before collect as well as learned information from subsystems of states with inputs, the exploration sound is first added to define the actual control input.[18] This manuscript explores the global control error for a nonlinear system class thru output feedback. The system nonlinearity unknown growth rate meets the homogeneous growth state. First, a homogeneous observer is built to assess the system position. In a power integrator method, a new dynamic output feedback controller is built to solve the problem of adaptive output control for the system (1) subject to a homogeneous growth state through an unknown growth rate. This method does not provide security against Brute-force attacks.[19]

The problem of the previous technique is that DoS attack, Brute-force attacks, and Brute-force dictionary attacks received access to the system, but did not carry out any activates after they succeeded. More probably it has been performed through automatic tools. Attackers carry out activates after they succeeded in break into the system. Further probably step has achieved through human. These attacks are malicious and interrupt the access to a computer network. The result of DoS attack can cause the computers on the network to squander their resources to serve illegitimate requests that result in a disruption of the network's services to legitimate users. With a sophisticated DoS attack, it becomes difficult to distinguish malicious requests from legitimate requests. Since a network layer DoS attack can cause interruptions to a network while causing collateral damage, it is vital to understand the measures to militate against such attacks.

The proposed Deep Adaptive Reinforcement Learning for Honeypot (DARLH) system contributes to protected honeypot implementation against DoS attacks. This DARLH system monitors both internal and external attacks using multilevel deep learning (DL) techniques. It uses protected poison distribution for an event management system. In addition, it extends the contribution of implement, event distribution and supervision practices, and DARLHs models (Level-1 and Level-2) creation. In Level-1, the proposed DARLHs system creates secure deep RL networks for monitoring both events and clients. In the next level, the system creates Deep Recurrent Neural Network (DRNN) and Deep Adaptive Reinforcement Learning (DARL) IDS agents integration modules for effective runtime attack detections. The Knowledge Data Discovery (KDD) data set pattern is used to DoS attack scenario.

The main contributions are given below:

- The problem of the previous technique is that the DoS attack is a malicious act with the goal of interrupting the access to a computer network. The result of the DoS attack can cause the computers on the network to squander their resources to serve illegitimate requests that result in a disruption of the network's services to legitimate users. To overcome this challenge this method is proposed.
- In this manuscript, the DARLH is proposed to observe the internal and external DoS attacks.
- The DARLH system executed DARL depends on IDS agents and DRNN depends on IDS agents for observing multiruntime DoS attacks.
- This system creates secure deep RL networks for monitoring both events and clients.
- Then the system creates DRNN and DARL IDS agent integration modules for effective runtime attack detections.
- To overcome the intrusion attacks, such as External DoS Attack, Internal DoS attack, Brute-force attack, DoS attack, Web attack, and Botnet attack, proposed method DARLH is used.
- Evaluation metrics are detection rate (DR), false alarm rate (FAR), and accuracy (ACC) is required for true and wrong classification.
- Here, the KDD data set pattern, UNSW-NB20, and Bot-IoT data sets are used to the scenario of DoS attack.
- The experimental outcomes are likened through different existing methods, such as GNBH, BCH, and RNSG.
- Here, the method is implemented in Python 3.7 as well as assessed performance. The experimental outcomes are likened through different existing methods, such as GNBH, BCH, and RNSG.

The remaining section of this manuscript is designed as follows: Section 2 delineates the related works and their motivation. Section 3 illustrates the proposed DARLHs system. Section 4 demonstrates the experimental results and discussion. Finally, Section 5 concludes the manuscript.

2 | RELATED WORKS

Honeypot implementation and security provisioning play a key role in any enterprise network systems. Even though many attack detection procedures emerge around the world, the DL-based honeypot implementation is really needful to construct dynamic NIDS agents. Attacks can inflict a variety of attacks that can be detrimental to network performance. Attackers are those who steal or harm the resources. The attacks, like, DoS, Spoofing, Wormhole, Black hole, and identity theft, are known as serious attacks in the network world. Among the attacks, DoS is an active attack category which degrades overall network performance.

Many research works contributed to DoS detection procedures and honeypot security schemes. They delivered various designs and implementation setups on detecting DoS in different network natures. Al-Nafjan et al.[6] presented an automated honeypot system for providing security features. Here, focused on automated network administration strategies in honeypot and honeynet are too similar to real systems, making it difficult to detect with a single feature. The system has multiple Local Area Networks (LANs) and resources with several attackers. The attacker events and real-time activities of honeypot unit were monitored by systematic procedures. The presented work performs base-level-automated functions in honeypot scenario without making any ML or DL systems.

Thu[20] established IDS agents and Intrusion Prevention System (IPS) implementations for cloud-based honeypot systems. The system mainly focused on malicious activities and malicious users in the cloud honeypot environment. Moreover, a new type of the honeypot system was implemented which diverts malicious attacks through different isolated paths. However, it was not more active against runtime attacks and it has lack of deep attack analysis techniques. The benefits of the method are 100% detect accuracy for limited attacks in the database. The disadvantage was zero-day and rule out attacks cannot be detected.

Banday and Sheikh[21] suggested captcha and timestamp-based honeypot security techniques. Here, it explored about text-based security provisions for protecting honeypots. Moreover, the data mining and data processing techniques were used to evaluate temporal data features. The advantages of the method are low overhead and computational cost. The disadvantage was little differences in attacks signature can simply bypass IDS agents.

Singh et al.[22] introduced fingerprint evaluation schemes for detecting DoS attacks in networks. Here, fingerprint detection techniques were evaluated using image processing procedures for identifying malicious activities. Both techniques were performing better in attack detection with conventional properties. The advantages of the method are small differences in attacks signature can simply bypass IDS agents. The disadvantage was High False Positive Rate.

Tsiropoulou et al.[23] have presented Mitigation of Interference Imposed through Intruders in networks. Here, a distributed iterative and low-complexity algorithm was presented. The advantages of the method are IDS agents' detection accuracy improves with time. The drawback was that detection was affected in the factors presence, such as low transfer power, collision, partial packet drop, false misbehavior, ambiguous collision, and so forth.

Vamvakas et al.[24] have presented a novel resource management framework to confirm efficient, wireless network smooth operation, supported through an unmanned aerial vehicle

(UAV), operational below nonorthogonal multiple access (NOMA) process, normal with malicious risk-awareness users. The advantages of this method are the 100% detection of high DR. The disadvantage was it consumes more energy.

Shrivastava and Hota[25] and Shi et al.[26] have introduced DoS attacks honeypot systems. Initially, the game theory and Naïve-Bayes techniques were used for detecting DoS attacks at runtime. Then the system used ML logics in finding DoS using training sets. It used basic ML techniques, not DL approaches. The block chain-based DoS attack detection in honeypot systems was introduced. Here, the technique was executed in distributed network environments. The introduced method shows effective performance in terms of security. But, the technique was not suitable for more active or centralized systems. The advantages of the method are capable of detecting known and unknown attacks. The disadvantage was the High Computational Cost.

Kaur and Singh[27] suggested a DRNN-based signature generation procedure to find attacks in honeypots. As DRNN was an effective Deep Neural Network (DNN), it creates more complex signatures for all events raised in honeypots. At the same time, this study implemented DRNN for signature generation procedures only. Here, DRNN could be extended for monitoring the events and attacks. Several techniques provided well-defined ML and DL-based attack detection procedures in honeypot systems. Yet they assumed the internal events were legitimate. This leads to internal honeypots. Particularly, internal DoS attacks were complicated to be identified and isolated. The method that benefits the database contains 100% detection accuracy of defined attacks. Disadvantage zero-day and rule cannot be detected outside of attacks.

Khan et al.[28] established multilevel botnet detection techniques using ML classifiers, such as Naïve Bayes, SVM, and Self-Organizing Maps. Here, the network domains, web events, and the conversations of network elements were evaluated. The advantages of this method are detecting capable dual zero-day, known attacks. The disadvantage was a defect that adds an additional module to the detection structure, which may lead to an additional delay in its improper hiring, a lower detection.

Srihari Rao et al.[29] and Sharma and Kaul[30] have presented various security techniques and DoS types. Here, vast network principles, attack problems, and cybercrimes in cloud-based systems were delivered. In addition, both works provide useful information about attack nature and motives at various vulnerable points of the network. The information gathered from these works was really useful for designing the proposed system. The advantages of the method are high DR. The disadvantage was that degrades the increase in high mobility of efficiency and performance.

Huang and Huang[31] designed the system using RL-based IDS agents. The system presents adaptive honeypot principles using RL and mark-over process structures. It gives a more cost-effective honeypot management system in terms of Quality of Service (QoS). However, the system dealt with QoS factors not with security credentials. At the same time, the designed system used the ML approach, not DL methodologies. The advantages of the method are low to no cost of maintenance as IDS agents learn, train itself utilizing network behavior, and network profiles created. The disadvantage was the high overhead and computational cost.

Kotey et al.[32] clarified various attack schemes and procedures that affect network resources. The method benefits the database that contains 100% detection accuracy of defined attacks. The disadvantage was zero-day and the rule attacks could not be detected out. From the vast analysis of related research work, the proposed system finds the lack of DL adaptations in runtime DoS detection. The proposed system solves the issues and complexities in the DL-based honeypot security system.

2.1 | Motivation of the research work

The existing techniques provide optimal support to honeypot, they lack complex activity validations. An intelligent attacker can perform some decent cryptanalysis techniques and compromised activities at the exposed point. These ML-based honeypots are not monitoring the changes inside the network. This is taken as a research problem which motivates the proposed system to develop deep RL engines to build bidirectional honeypots. Therefore, the RL enabled bidirectional honeypots monitor both internal and external attackers. In this regard, these honeypot systems are implemented in Virtual Local Area Network (VLAN) which seems to be a directly connected physical LAN. It provides all types of data that look like legitimate to both internal and external users. By this trap, the proposed honeypots protect the strong security layer between the data and the attackers.

3 | DEEP ADAPTIVE REINFORCEMENT LEARNING FOR HONEYPOTS SYSTEM

The proposed DARLH system is designed and executed to observe both internal and external attackers. This attacker may inject a DoS attack into any system deployed in a honeypot environment. This environment has one or more monitoring resources (Servers and Clients), which are enabled with active IDS agents. The IDS agent is running inside the monitoring resources to scrutinize each and every network activity (traffics). On the basis of the network condition or screen resolution, the client decides which media segment should be downloaded. The signaling data of DASH are characterized as content-level information, period-level information, adaptation information, and mapping information. Due to heterogeneous nature of the mobile network and the unpredictable channel with optimization in terms of resource and technology are useful. The proposed DARLHs contain the following technical aspects for building more reactive honeypots.

- Secure Honeypot Traffic Distribution Model,
- Server Distribution and Supervising Model,
- Honeypot with DARLHs Level-1 Model,
- Honeypot with DARLHs Level-2 Model.

The secure honeypot traffic model helps distribute the events from the server to parallel clients.

Clients monitor the events and send reports to the server. The server supervising model is used to monitor the client's activities. DARLH Level-1 and DARLH Level-2 models are implemented for carrying DoS attack detection activities using DARL and Deep Recurrent Neural NIDS agents. To analyze the complexity of the network, first makes the followed definitions. Let Nfb denotes the count of frequent traffic found, Nd denotes the total count of dimensions, and $Nd(fb)$ denotes the total count of dimensions. In other words, if there are n dimensions in feature space then frequent baskets are founded in m dimensions, then $Nd(fb) = m$. Consider $Nd \geq Nd(fb)$. Here, μ as the average number of frequent traffic, so $\mu = NfbNd(fb)$ and $\mu \geq 1$.

3.1 | Secure honeypot traffic distribution model

The proposed honeypot traffic model is implemented for building traffic collection methodology. Honeypot environment contains a database, database server, communication devices (switches and routers), and monitoring nodes. The monitoring nodes are the collection of servers and client machines. Client monitoring machines run their IDS agents unit to monitor the activities injected from any side of the network. The IDS has been equipped with the proposed procedures. The attacker database is attached with each client machine independently to make attacker identification. The reports and alert messages are delivered from client machines to servers.

The monitoring servers evaluate the reports delivered from various monitoring nodes. The servers in the honeypot are implemented with supervised IDS agents. This agent monitors the activities of client machines and the attack reports. In addition, the network traffic is injected into honeypot queues.[33,34] Then, the server distributes the network traffics among the clients connected to it. The honeypot traffic model follows the authentic poison distribution model and the protected queuing model.

Definition 1. In Honeypot, H (P) contains n similar-monitoring servers and m client machines. In this system, n server collects the network events at the rate T^e and R^μ generates the attack reports at the rate U^i. At time τ, the monitoring resources of the honeypot are utilized by the events at the rate U^i. As the client machines are connected to servers, U^i implies the aggregated utilization rate. Equations (1) and (2) illustrate traffic models and utilization factors, respectively.

$$U^i = \frac{T^e}{R^\mu n}, \tag{1}$$

$$U^i = S^i + C^i, \tag{2}$$

where

U^i—honeypot utilization factor,
T^e—average traffic rate at honeypot event queues,
R^μ—average response rate of each server,
n—total number of monitoring servers,
S^i—utilization factor of honeypot server,
C^i—utilization factor of honeypot client machines,

$$C^i = C^1 + C^2 + \cdots + C^n. \tag{3}$$

Equation (3) denotes the individual utilization rate of honeypot clients. In each client machine, traffic queues are available to get the network events continuously. There are two types of events queues implemented inside each machine. This queue setup is common to servers and client machines. The types of queues are given below,

- Internal Traffic Queues,
- External Traffic Queues.

In each server traffic-queue (internal or external), eXn traffic events wait for server response. At the same time, server distributes the traffic events to the client machines. In each client traffic-queue (internal or external), e^iXm^i traffic events wait for the completion of monitoring process, m^i denotes the client machine identifiers. e^i denotes the distributed network events to each client machine in honeypot.

In this case, $Q^{\mathrm{SI}} \gg Q^{e\mathrm{I}}$ and $Q^{\mathrm{SE}} \gg Q^{e\mathrm{E}}$, let take on $Q^{e\mathrm{I}}$ and $Q^{e\mathrm{E}}$, the number of internal and external traffic queues in each client machine, respectively. The number of internal and external traffic queues in each monitoring server indicates Q^{SI} and Q^{SE}, respectively. Now, the honeypot traffic arrival and distribution models are given in Equation (4). The honeypot poison model is determined as follows:

$$H^{\mathrm{P}} = e^{-\tau^e}\frac{(\tau^e)^l}{n!} \pm b, \tag{4}$$

where

l—total number of traffic event arrivals,

b—event behavior factor.

The poison model used for honeypot traffic distribution uses secure procedures which are incorporated in IDS agent modules. Algorithm 1 describes the credentials-based poison distribution. In this technique, the events and the sources are validated before the event distribution. This kind of validation is completed at honeypot monitoring servers.

In this honeypot circumstance, the servers and clients maintain multiple parallel queues. The queuing processes are defined by the honeypot multisystem queuing model, $M/M/n$, $n > \tau^e$. This is determined for all the finite number of internal and external traffic events. Therefore, the queuing model becomes, $M/M/n/\infty/l$ with the rate of (eXn) at servers. At the same time, this rate is determined the clients as e^iXm^i. The honeypot traffic distribution model is used to distribute the internal and external events into honeypot servers and clients, respectively. Section 3.2 describes about the server distribution and supervising model.

Algorithm 1: Honeypot traffic handling using authentic poison distribution

```
Input: Traffic events (internal and external)
Output: Received Events at Client Queues
Begin
Step 1: Get events, REQ ($e^{\mathrm{ID}}$, $e^{\mathrm{RID}}$, $S^{\mathrm{ID}}$)
$e^{\mathrm{ID}}$—event identifier, $e^{\mathrm{RID}}$—event request identifier, $S^{\mathrm{ID}}$—event source credentials
$object^{\mathrm{ID}} \leftarrow$ Exact object id from $S$
Timestamp $\leftarrow$ exact timestamp from $S$
$state^{\mathrm{ID}} \leftarrow$ Exact state id from $S$
Emit $(object^{\mathrm{ID}}, (timestamp, state^{\mathrm{ID}}, S))$
Function: REDUCE preprocess $(object^{\mathrm{ID}}, (timestamp, state^{\mathrm{ID}}, S))$
Step 2: Execute the procedure, $H^{\mathrm{P}}$ for all valid events
Step 3: Repeat $H^{\mathrm{P}}$ for all $Q^{\mathrm{SI}}$ and $Q^{\mathrm{SE}}$
End
```

3.2 | Server distribution and supervising model

This model describes secure traffic distribution and client supervising tasks executed at server units. As discussed, there are n supervising servers available for monitoring m clients. The IDS agents enabled client machines are deployed for monitoring the events distributed from n series. An authentic poison distribution model helps share the events among various client machines. To deliver the events to clients, the server verifies the credentials of client machines and other resources.

Definition 2. A Honeypot gets continuous events from internal and external participants at time τ^i. The internal and external events are denoted as e^{I} and e^{E}, respectively. The events are collected at multiple Q^{SI} and Q^{SE} at regular intervals. The gathered events are distributed to $Q^{e\mathrm{I}}$ and $Q^{e\mathrm{E}}$. The total events delivered to each queue as e^i. Honeypot servers validate m^i client credentials before the event distribution process. Algorithm 2 denotes server distribution and supervising model procedures.

The secret shared key communication between the clients and servers is enabled using shared secret key S^K.

$$S^{\mathrm{KH}} = SHA^{512}(S^{\mathrm{IP}} \,|(C^{\mathrm{IP}}||T||N). \tag{5}$$

Algorithm 2: Distribution and supervision model

```
Input: $e^{\mathrm{I}}$ and $e^{\mathrm{E}}$
Output: Events at $Q^{e\mathrm{I}}$ and $Q^{e\mathrm{E}}$
Begin
Step 1: Get internal and external events, $e^{\mathrm{I}}$ and $e^{\mathrm{E}}$ at $Q^{\mathrm{SI}}$ and $Q^{\mathrm{SE}}$, respectively
Step 2: Check for client monitoring requests, ReQ ($C^{\mathrm{IP}}$, $T^{\mathrm{C}}$, $S^{\mathrm{K}}$, $C^{\mathrm{P}}$, $C^{\mathrm{T}}$)
  $C^{\mathrm{IP}}$–Client IP address, $T^{\mathrm{C}}$—Client TCP credentials, $S^{\mathrm{KH}}$—Secret SHA key (512 bits)
  $C^{\mathrm{P}}$—Honeypot client policy credentials and $C^{\mathrm{T}}$—request timestamp
Step 3: If ReQs available in client request queues
  Then validate ReQ for all credential
Step 4: Server sends a request, ReQS ($S^{\mathrm{IP}}$, $T^{\mathrm{S}}$, $S^{\mathrm{K}}$, $S^{\mathrm{P}}$, $S^{\mathrm{T}}$, $Q^{\mathrm{A}}$)
$S^{\mathrm{IP}}$—Client IP address, $T^{\mathrm{S}}$—Server TCP credentials, $S^{\mathrm{KH}}$—Same secret SHA key (512 bits)
$S^{\mathrm{P}}$—Honeypot server policy credentials, $S^{\mathrm{T}}$—Request timestamp, $Q^{\mathrm{A}}$—Status ($Q^{e\mathrm{I}}$ and $Q^{e\mathrm{E}}$)
Step 5: Client gives response to server with $Q^{\mathrm{A}}$ status
Step 6: Call $H^{\mathrm{P}}$ based on $Q^{\mathrm{A}}$ status
Step 7: Complete the event distribution
  That is, $Q^{\mathrm{SI}}$ to $Q^{e\mathrm{I}}$ and $Q^{\mathrm{SE}}$ to $Q^{e\mathrm{E}}$
Step 8: Enable Client Supervision Mode (CSM)
Step 9: Request the clients to put them in Promiscuous Mode (PRM)
End
```

The CSM helps the servers to monitor their associated clients when the clients put themselves in open promiscuous mode for appropriate servers. Servers are attached with the main attacker database (DoS) and the client activity control system. The client activity control system validates each and every action of client machines based on preconfigured rules (rule-based classification). This helps identify and supervise the legitimate activities of client machines. In addition, the server keep master DoS database (KDD Data set) in the safe zone for supervising fault DRs.

3.3 | Honeypots with DARLHs Level-1 model

DARLHs functions are implemented using DARL techniques. DARL approaches are designed with the help of event-based reward functions as portrayed in Figure 1.

Figure 1 demonstrates the DARL-based IDS agents execute actions against DoS attacks injected in honeypot environment. The environment provides rewards (positive or negative) for each action taken by IDS agents. On the basis of the statewise rewards provided to IDS agent, it trains itself to handle next action against DoS. In this figure both internal and external attackers inject DoS attacks into honeypot environment.[35] Figure 2 shows the Specific Event Distribution and Client Control Model

The servers located in the monitoring section of honeypot distribute the events to clients as described in Algorithms 1 and 2. In case, the attack is effectively detected, the environment generates positive reward to DARL IDS agent. Otherwise, it generates negative notifications. Algorithm 3 gives DARLHs Level-1 functions and DARL agent activities in honeypot environment. In the environment, DoS monitoring system, data resources, and other devices are deployed.

```
**Algorithm 3:** DARLH Level-1 model
**Input:** Events from $Q^{eI}$ and $Q^{eE}$
**Output:** DARL action on DoS Events
**Begin**
**Step 1**: Set client at PRM and delivers, $P^{T}$, the traffic port to server
**Step 2**: Activate DARL agent function in each client machine, $m^{i}$
**Step 3**: Determine $Q^{eI}$ and $Q^{eE}$ sizes with waiting events in queue
**Step 4**: Construct DNN for evaluating $e^{I}$ and $e^{E}$
**Step5**: Determine internal and external event functions $m^{i}IE_{(Q)}$ and $m^{i}EE_{(Q)}$, respectively

    $$m^{i}IE_{(Q)} = Q_0(States, e^{I}, T^{S}) \tag{6}$$
    $$m^{i}EE_{(Q)} = Q_0(States, e^{E}, T^{S}) \tag{7}$$

**Step 6:** Call procedure DoS (Data items_KDD) for all $e^{I}$ and $e^{E}$ on various states
**Step 7:** Do match function DARL(DoS)
If (($m^{i}IE_{(Q)}$&&$m^{i}EE_{(Q)}$)==$KDD^{DoS}$)
Then set action as "Alert"
Else action as "Allow"
**Step 8**: Compute internal and external RL action function.

    $$m^{i}IA_{(Q)} = Q_0(Action \quad States, e^{I}, a, T^{S}) \tag{8}$$
    $$m^{i}EA_{(Q)} = Q_0(Action \quad States, e^{E}, a, T^{S}) \tag{9}$$

**Step 9:** Store the deep $Q$ values for all $m^{i}IA_{(Q)}$ and $m^{i}EA_{(Q)}$
**Step 10:** Create internal and external alert reports, $Report^{IA}$ and $Report^{EA}$ for all $m^{i}$
**Step 11:** Deliver, $Report^{IA}$ and $Report^{EA}$ to servers
**Step 12:** Server generates rewards to each action at different states and sends to clients
**Step 13**: Clients' RL Intrusion Detection System agent receives reward, $R^{S}$.
**Step 14:** Train the RL Intrusion Detection System agent function depend on received $R^{S}$
End
```

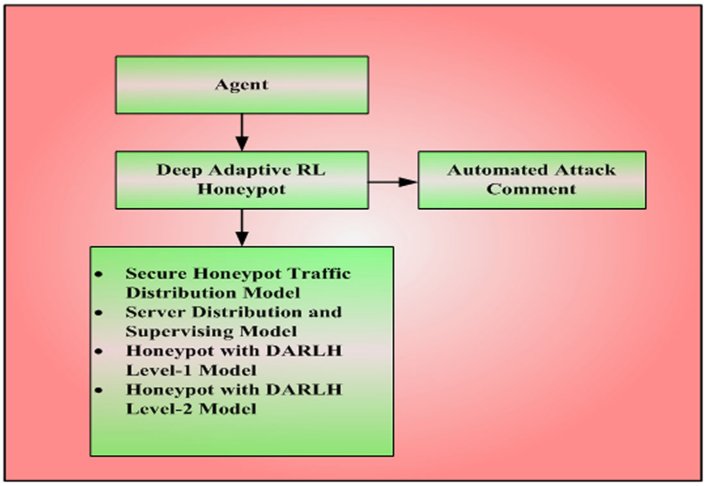
Agent
Deep Adaptive RL Honeypot
Automated Attack Comment
• Secure Honeypot Traffic Distribution Model
• Server Distribution and Supervising Model
• Honeypot with DARLH Level-1 Model
• Honeypot with DARLH Level-2 Model

FIGURE 1 Framework of the DARLH system. DARLH, Deep Adaptive Reinforcement Learning for Honeypot; RL, reinforcement learning [Color figure can be viewed at wileyonlinelibrary.com]

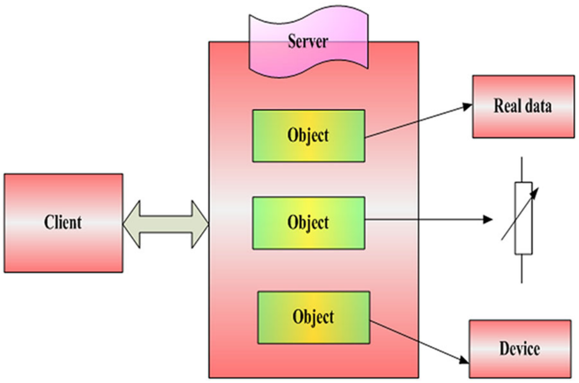

FIGURE 2 Specific event distribution and client control model [Color figure can be viewed at wileyonlinelibrary.com]

Figure 3 shows the DARLH Level-2 Model. The DARLH agent runs in each client machine. This agent follows deep RL structure that contains multiple layers of hidden layers and complex event analysis functions to detect DoS. In addition, the DoS detection results are forwarded to supervisor server units deployed in honeypot environment. Particularly, these deep IDS agents follow the Markov process for finding sequential observations. In this proposed system, the deep Q values are used to specify each event's particulars. At the same time, the events are predicted using current Q values of events, which are shown in Equation (10).

$$EC_{i(Q+1)}(S_i, a_\tau) = EC_{(Q)}(S_i, a_\tau) + \alpha(S_i, a_\tau) \times \{(ma^{S_i} + mi^{S_i}) + R^{S}\}. \tag{10}$$

here $EC_{i(Q+1)}(S_i, a_\tau)$ predictable network events at the state S_i with an action at the time a_τ. $EC_{(Q)}(S_i, a_\tau)$ and R^S represent current event particulars and rewards, respectively. In addition, this RL-based DoS detection uses learning rate at each state $\alpha(S_i, a_\tau)$.

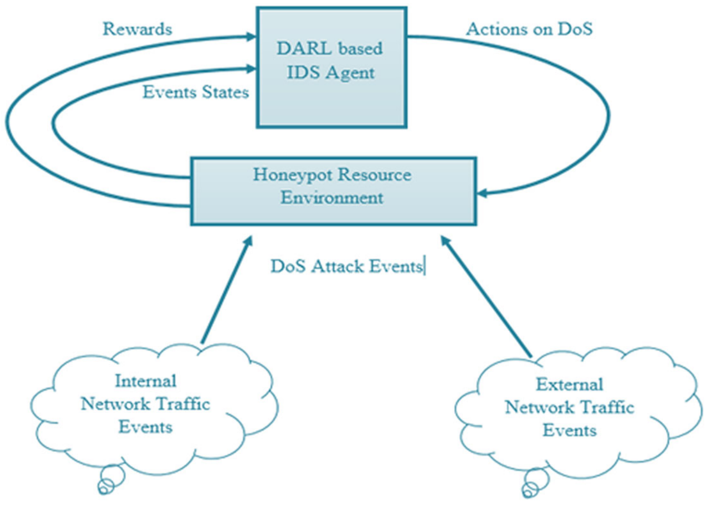
Rewards
Actions on DoS
DARL based IDS Agent
Events States
Honeypot Resource Environment
DoS Attack Events
Internal Network Traffic Events
External Network Traffic Events

FIGURE 3 DARLH Level-2 model. DARL, Deep Adaptive Reinforcement Learning; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; IDS, Intrusion Detection System [Color figure can be viewed at wileyonlinelibrary.com]

$$R^{S} = R^{S}{}_{(Q+1)} \pm \omega - EC_{(Q)}(S_i, a_\tau). \tag{11}$$

Equations (11) and (12) describe reward functions and the action taken after getting a server-side reward, ω specifies the reward variation factor. In Equation (12), $Q(e)_O$ is event action on reward and σ is an action bias.

$$Q(e)_O = \sum_{i=1}^{L} f(R^S \pm \sigma). \tag{12}$$

The DARL IDS agent functions are effectively utilizing the server reward function and event attributes for finding DoS patterns. At the same time, this DARL has well-trained DNN and deep Q values. However, the DARLHs Level-1 model uses only DARL IDS agent against dynamic DoS attacks. This leads a technical problem for more dynamic events. To solve this issue, the proposed method implements the DARLH Level-2 mechanism.

3.4 | Honeypots with DARLH Level-2 models

DARLH Level-2 mechanism helps integrate both DRNN and DARL-based IDS agents to monitor DoS attacks in honeypot. The dual DNNs are connected in a pipeline manner to accomplish more accurate detections. In this case, both networks are feedback structured networks. These dual DNNs are implemented with IDS agent of the honeypot system. Figure 4 depicts the DNNs connectivity of DARLHs Level-2 Model. Though DARL supports for dynamic arrival of network event, it lacks at more uncertain conditions. To solve this problem DRNN is required against real-time uncertainties.

Algorithm 4 illustrates DRNN and DARL combinations for detecting DoS attacks in honeypot. Here, DRNN is a Level-1 DNN and DARL is a Level-2 DNN. DRNN generates the decisions on DoS using KDD training sets and the rule-based classifications. The events and

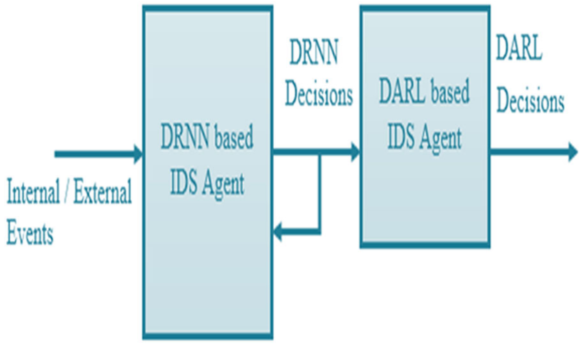

FIGURE 4 Deep adaptive reinforcement learning for honeypots Level-2 model. DARL, Deep Adaptive Reinforcement Learning; DRNN, Deep Recurrent Neural Network; IDS, Intrusion Detection System [Color figure can be viewed at wileyonlinelibrary.com]

results generated by DRNN are evaluated with the help of DARL IDS agents. Both IDS agents are using KDD DoS features and real-time policy irregularities happen due to attacks. DRNN and DARL networks are trained themselves in each iteration of honeypot events. In addition, the dual DL techniques support implementing complex attack analysis processes.

Algorithm 4 describes the steps involved in DRNN and DARL networks to detect DoS attack detection. The proposed honeypot environment creates more complex DL techniques for enquiring both internal and external DoS attacks. In this regard, it deals with separate event queues in client and server units. In the proposed DARLHs system, client machines are monitoring DoS events on both sides parallel. The event distribution and reward management functions are controlled by server units. This is a completely protected environment against DoS attacks. Section 4 describes implementation details and results produced for evaluating the performance of the proposed system.

Algorithm 4: DARLH Level-2 model

```
Input: Events from $Q^{eI}$ and $Q^{eE}$
Output: DARLH Level-2 action on DoS Events
Begin
Step 1: Call event initialization steps and functions in $Q^{eI}$ and $Q^{eE}$
Equations (6) and (7)
Step 2: Call procedure DoS (Data items_KDD) for all $e^{I}$ and $e^{E}$ on various states
Step 3: Compute, Recurrent event observations
                    $e_T = \tan e(\omega_{ee} \cdot e_{T-1}, \omega^{e})$          (13)
  $\omega_{ee}$—RNN weight function, $e_{T-1}$—Previous observations, and $\omega^{e}$—Input event weight
Step 4: Construct Deep Recurrent Neural Network using hidden weight functions
Do match function Deep Recurrent Neural Network (Denial of Service)
If $((m^{i}IE_{(Q)}\&\&m^{i}EE_{(Q)})==KDD^{\mathrm{DoS}})$
Then set action as "Alert"
Else action as "Allow"
Step 5: Collect and store RNN observations as Level-1 events
Step 6: Call DARLH Level-1 procedures (Algorithm 3)
End
```

4 | EXPERIMENTAL SETUP AND RESULTS

An intelligent attacker can execute several decent cryptanalysis techniques and compromised activities. The changes inside the network may not be monitored by these ML-based honeypots. The proposed system contributes to protected honeypot implementation against attacks of DoS. The proposed DARLH system is executed with 10 server units and a generic database. Each server in the honeypot controls 25 client machines. The proposed DARL and DRNN IDS agents are implemented in servers and clients. At the same time, the event distribution model and client control principles are deployed in servers only. For detecting DoS attacks, the IDS agents run rule-based classification procedures and signature mapping techniques. In this environment, the KDD′ 99, UNSW-NB20, and Bot-IoT data sets are deployed at server points and client points. Server points have the main data set but client machines have training data sets and test data sets.

4.1 | Key parameter

Selecting maximum suitable parameters for the model is the model training necessary part. The random forest algorithm that determines the accuracy of model classification has two main parameters. They are "*n-estimators*" and "*max-depth*". Here, "*n-estimators*" is the count of DRNN as well as "*max-depth*" is the maximal permissible depth of every iteration. If "*n-estimators*" is too big, outcomes possibly over fitting. However, if "*n-estimators*" is too small, outcomes possibly under fitting. So, an "*n-estimators*" suitable value for the predicted final model classification accuracy is important. {100, 120, 200, and 300} is ready for "*n-estimators*", {96.2 to 100} ready for "*max-depth*". It tested these two collections separately to choose the most suitable value for these two parameters. Table 1 shows the simulation parameter.

By using Weka 3.0 and Python 3.7 tools, this experiment is generated. From three data sets, unwanted data are removed the missing data from original data set. To implement the proposed DARLHs technique Python 3.7 is used. The performance of the proposed system is evaluated using metrics, such as DR, FAR, true negative rate (TNR), and ACC. This is evaluated for both internal and external DoS attacks. Then the results are compared with the existing schemes, like, GNBH, BCH, and RNSG.

GNBH technique uses game-theoretical schemes and Naïve-Bayes schemes for detecting DoS attacks in honeypot systems. From the overall events this technique classifies and isolates DoS. Here, the ML techniques are used to delivered significant outcomes Out. Moreover, the techniques are conventional tactics against runtime attacks, like, DoS. In the same manner, BCH is used to construct a secure and distributed honeypot system. To provide authentic transaction in community networks block chain technology is used. The complicated cryptography techniques are used to protect the data and also building protected honeypot systems. This method does not evaluate attack scenarios in the experimental setup. However, this system evaluated security aspects in the performance evaluation.

RNSG is a DL-based signature generation system to protect the network against attacks. A complicated signature pair provides more security in network transactions. The signature credentials are created using DRNN structures for detecting intrusions. This system utilized signature-based pattern recognition techniques for intrusion detection, also implemented signature generation and pattern recognition techniques using DRNN. This is an effective technique for detecting intrusion detection. But, it has limitations related to runtime dynamics, such as traffic variations and parallel monitoring complications.

4.2 | Data set description

In this method three types of data set are organized at server points and client's points. Server points have the main dataset but client machines have training datasets and test datasets. for example, KDD data set, UNSW-NB20, and Bot-IoT data sets.

4.2.1 | KDD′ 99 data set

The KDD′ 99 data set is created by Cyber Range Lab of Australian Centre for Cyber Security (ACCS).[36] A partition of the full data set is provided, separated into a training set, a test set. For training 125,920 records are taken and 22,342 records for testing with a total of 148,262 records. The number of features is 43 with the class label. Different attacks that are present in KDD′ 99 are shown in Table 2.

4.2.2 | UNSW-NB20 data set

This data set is created by Cyber Range Lab of ACCS.[37] It represents new modern normal activities containing contemporary attacks. A full data set partition is provided, separated into a training set, a test set. For training 83,057 records are taken and 20,764 records for testing with a total of 103,821 records. The number of features is 43 with the class label. There are 10 categories in total, one for normal class representing no attacks and nine attacks: shellcode, backdoor, exploits, worms, reconnaissance, generic, analysis, DoS, and fuzzers. This data set is more complex than KDD′ 99 because it contains features where the attacks and normal classes have similar behaviors. Different attacks that are present in UNSW-NB20 are shown in Table 3.

4.2.3 | Bot-IoT data set

The Bot-IoT data set is introduced by Cyber Range Lab of Centre of UNSW Canberra Cyber.[38] The main characteristic of this data set is that it represents a realistic network environment with more attacks and network traffic in a realistic setting with its respective labels. The data set has normal IoT-related as well as other network traffic, with several categories of attack traffic commonly utilized through botnets. The attack categories in the data set include Key logging, Data exfiltration, DDoS, DoS, OS, and Service Scan. The dataset has 4 components: network platforms, simulated IoT services, extracting features and forensics analytics. For training 1,149,388 records are taken and 639,205 records for testing with a total of 1,788,593 records. Different attacks that are present in Bot-IoT data set are shown in Table 4.

4.3 | Performance metrics

The performance metrics of TNR, DR, FAR, and ACC are required for true and wrong classification.

TABLE 1 Simulation parameter

Simulation parameters	Values
Simulation area	(800×800) m
No. of nodes	100, 200, 300
Nodes speed	1 and 15 m/s
Power levels	3 and 7 dBm
Simulation time	400 s
Time of simulation	2000 s
Reporting time (T_r)	26 s
Sampling time (T_s)	5 s
Mobility model	RWP, GM
Traffic rate (normal/attack)	200/2048 kbps
Number of traffic	$l = 386$
Dimension *CM*	100
Attacks	1
Data set used	3

$$Accuracy = \frac{TP_{\text{Attack}} + TN_{\text{Benign}}}{TP_{\text{Attack}} + FN_{\text{Attack}} + TN_{\text{Benign}} + FP_{\text{Benign}}},$$

$$DR_{\text{Attack}} = \frac{TN_{\text{Benign}}}{TN_{\text{Benign}} + FP_{\text{Benign}}},$$

$$TNR_{\text{Benign}} = \frac{FP_{\text{Benign}}}{TN_{\text{Benign}} + FP_{\text{Benign}}},$$

$$FAR = \frac{FP_{\text{Benign}}}{TN_{\text{Benign}} + FP_{\text{Benign}}}.$$

4.3.1 | Performance comparison of various data sets

The performance metrics are DR, FAR, TNR, and ACC of various data sets are given below:

(a) *KDD′ 99 data set*

Tables 5 and 6 show the ACC, FAR, DR, TNR of various methods.

Table 5 shows the ACC and FAR of UNSW-NB20 Data set. In this the proposed method shows high accuracy while comparing to other methods, such as GNBH, BCH, and RNSG. In Brute-force attack the accuracy of the proposed DARLH is 76.11%, 73.11%, and 74.11% higher than that of the existing methods. In DoS attack the accuracy of the proposed DARLH is 86.11%, 73.11%, and 64.11% higher than that of the existing methods. In Web attack the

TABLE 2 Attack types in KDD' 99 data set

Category	Attack type	Flow count	Training	Test
Brute force	SSH—Brute force	232	995	2754
DoS attack	DoS attacks—Slow HTTP test	13,987	45,927	7457
Web attack	Brute force—Web	18,336	67,342	9710
Botnet	Bot	27,619	11,656	2421

Abbreviations: DoS, Denial of Service; HTTP, Hypertext Transfer Protocol; KDD, Knowledge Data Discovery; SSH, secure shell.

TABLE 3 Attack types in UNSW-NB20 data set

Category	Attack type	Flow count	Training	Test
Brute force	SSH—Brute force	230	184	46
DoS attack	DoS attacks—Slow HTTP test	13,989	55,956	13,989
Web attack	Brute force—Web	19,336	15,469	3867
Botnet	Bot	28,619	11,448	2862

Abbreviations: DoS, Denial of Service; HTTP, Hypertext Transfer Protocol; SSH, secure shell.

TABLE 4 Attack types in Bot-IoT Data set

Category	Attack type	Flow count	Training	Test
Brute force	SSH—Brute force	9543	7634	1909
DoS attack	DoS attacks—Slow HTTP test	19,547,603	156,380	390,952
Web attack	Brute force—Web	12,315,997	985,280	246,320
Botnet	Bot	118	94	24

Abbreviations: DoS, Denial of Service; HTTP, Hypertext Transfer Protocol; SSH, secure shell.

TABLE 5 Accuracy and FAR of KDD' 99 data set

	Accuracy				False alarm rate			
Attacks methods	Brute force	DoS attack	Web attack	Botnet	Brute force	DoS attack	Web attack	Botnet
DARLH (proposed)	92.11	95.4	98.1	97.6	0.90	1.0	0.99	0.97
GNBH	80.4	81.3	72.4	74.1	0.32	0.43	0.65	0.54
BCH	82.1	69.2	70.6	70.45	0.35	0.65	0.76	0.42
RNSG	56.1	33.4	42.3	56.32	0.45	0.65	0.26	0.42

Abbreviations: BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; FAR, false alarm rate; GNBH, Game and Naïve-Bayes Honeypot; KDD, Knowledge Data Discovery; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection.

TABLE 6 TNR and DR of KDD' 99 data set

	True negative rate				Detection rate			
Attacks methods	Brute force	DoS attack	Web attack	Botnet	Brute force	DoS attack	Web attack	Botnet
DARLH (proposed)	0.94	0.93.4	0.92	0.90	0.91	1.0	0.95	0.96
GNBH	0.82	0.84.3	0.70	0.70	0.32	0.43	0.65	0.54
BCH	0.83	0.69.2	0.70	0.70	0.55	0.65	0.76	0.42
RNSG	0.79	0.69.4	0.88	0.60	0.55	0.65	0.26	0.42

Abbreviations: BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; DR, detection rate; GNBH, Game and Naïve-Bayes Honeypot; KDD, Knowledge Data Discovery; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection; TNR, true negative rate.

accuracy of the proposed DARLH 43.11%, 63.11%, and 56.11% is higher than that of the existing methods In Botnet attack the accuracy of the proposed DARLH is 66.11%, 53.11%, and 84.11% higher than that of the existing methods.

In Brute-force attack the FAR of the proposed DARLH is 6.11%, 3.11%, and 7.11% higher than that of the existing methods. In DoS attack the FAR of the proposed DARLH is 8.11%, 7.11%, and 6.11% higher than that of the existing methods In Web attack the FAR of the proposed DARLH 3.11%, 13.11%, and 15.11% is higher than that of the existing methods In Botnet attack the FAR of the proposed DARLH is 16.11%, 13.11%, and 14.11% higher than that of the existing methods.

Table 6 shows the DR and TNR of UNSW-NB20 Data set. In this the proposed method shows high TNR and DR while comparing to other methods. The existing methods, such as GNBH, BCH, and RNSG. In Brute-force attack the TNR of the proposed DARLH is 16.11%, 13.11%, and 9.11% higher than that of the existing methods. In DoS attack the TNR of the proposed DARLH is 8.11%, 7.11%, and 6.11% higher than that of the existing methods In Web attack the TNR of the proposed DARLH 13.11%, 3.11%, and 6.11% is higher than that of the existing methods. In Botnet attack the TNR of the proposed DARLH is 46.11%, 13.11%, and 14.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

In Brute-force attack the DR of the proposed DARLH is 6.11%, 3.11%, and 7.11% higher than that of the existing methods. In DoS attack the DR of the proposed DARLH is 8.11%, 7.11%, and 6.11% higher than that of the existing methods. In Web attack the DR of the proposed DARLH 3.11%, 13.11%, and 15.11% is higher than that of the existing methods. In Botnet attack the DR of the proposed DARLH is 16.11%, 13.11%, and 14.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

(b) *UNSW-NB20 Data set*

Tables 7 and 8 show the ACC, FAR, DR, TNR of various methods.

Table 7 shows the accuracy and FAR of UNSW-NB20 Data set. In this the proposed method shows high accuracy while comparing to other methods, such as GNBH, BCH, and RNSG. In Brute-force attack the accuracy of the proposed DARLH is 76.11%, 73.11%, and 74.11% higher than that of the existing methods. In DoS attack the accuracy of the proposed DARLH is 86.11%, 73.11%, and 64.11% higher than that of the existing methods. In Web attack the accuracy of the proposed DARLH 43.11%, 63.11%, and 56.11% is higher than that of the

TABLE 7 Accuracy and FAR of UNSW-NB20 data set

	Accuracy				False alarm rate			
Attacks methods	Brute force	DoS attack	Web attack	Botnet	Brute force	DoS attack	Web attack	Botnet
DARLH (proposed)	96.11	93.4	92.1	90.6	0.90	1.0	0.99	0.97
GNBH	82.4	84.3	70.4	70.1	0.32	0.43	0.65	0.54
BCH	83.1	69.2	70.6	70.45	0.55	0.65	0.76	0.42
RNSG	79.1	69.4	88.3	60.32	0.55	0.65	0.26	0.42

Abbreviations: BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; FAR, false alarm rate; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection.

TABLE 8 TNR and DR of UNSW-NB20 data set

	True negative rate				Detection rate			
Attacks methods	Brute force	DoS attack	Web attack	Botnet	Brute force	DoS attack	Web attack	Botnet
DARLH (proposed)	0.94	0.93.4	0.92	0.90	0.91	1.0	0.95	0.96
GNBH	0.82	0.84.3	0.70	0.70	0.32	0.43	0.65	0.54
BCH	0.83	0.69.2	0.70	0.70	0.55	0.65	0.76	0.42
RNSG	0.79	0.69.4	0.88	0.60	0.55	0.65	0.26	0.42

Abbreviations: BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; DR, detection rate; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection; TNR, true negative rate.

existing methods. In Botnet attack the accuracy of the proposed DARLH is 66.11%, 53.11%, and 84.11% higher than that of the existing methods.

In Brute-force attack the FAR of the proposed DARLH is 6.11%, 3.11%, and 7.11% higher than that of the existing methods. In DoS attack the FAR of the proposed DARLH is 8.11%, 7.11%, and 6.11% higher than that of the existing methods. In Web attack the FAR of the proposed DARLH 3.11%, 13.11%, and 15.11% is higher than that of the existing methods. In Botnet attack the FAR of the proposed DARLH is 16.11%, 13.11%, and 14.11% higher than that of the existing methods.

Table 8 shows the DR and TNR of UNSW-NB20 Data set. In this the proposed method shows high TNR and DR while comparing to other methods. The existing methods such as Game and Naïve Bayes Honey pot (GNBH), Block chain Honey pot (BCH) and RNN based Signature Generation and Detection (RNSG) respectively. In Brute-force attack the TNR of the proposed DARLH is 16.11%, 13.11%, and 9.11% higher than that of the existing methods. In DoS attack the TNR of the proposed DARLH is 8.11%, 7.11%, and 6.11% higher than that of the existing methods. In Web attack the TNR of the proposed DARLH 13.11%, 3.11%, and 6.11% is higher than that of the existing methods. In Botnet attack the TNR of the proposed DARLH is 46.11%, 13.11%, and 14.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

In Brute-force attack the DR of the proposed DARLH is 6.11%, 3.11%, and 7.11% higher than that of the existing methods. In DoS attack the DR of the proposed DARLH is 8.11%, 7.11%, and

6.11% higher than that of the existing methods. In Web attack the DR of the proposed DARLH 3.11%, 13.11%, and 15.11% is higher than that of the existing methods. In Botnet attack the DR of the proposed DARLH is 16.11%, 13.11%, and 14.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

(c) *Bot-IoT Data set*

Tables 9 and 10 show the ACC, FAR, DR, TNR of various methods.

Table 9 shows the accuracy and FAR of Bot-IoT Data set. In this the proposed method shows high accuracy while comparing to other methods, such as GNBH, BCH, and RNSG. In Brute-force attack the accuracy of the proposed DARLH is 76.11%, 73.11%, and 74.11% higher than that of the existing methods. In DoS attack the accuracy of the proposed DARLH is 86.11%, 73.11%, and 64.11% higher than that of the existing methods. In Web attack the accuracy of the proposed DARLH 43.11%, 63.11%, and 56.11% is higher than that of the existing methods. In Botnet attack the accuracy of the proposed DARLH is 66.11%, 53.11%, and 84.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

In Brute-force attack the FAR of the proposed DARLH is 5.11%, 4.11%, and 3.11% higher than that of the existing methods. In DoS attack the FAR of the proposed DARLH is 7.11%,

TABLE 9 Accuracy and FAR of Bot-IoT data set

	Accuracy				False alarm rate			
Attacks Methods	Brute force	DoS attack	Web attack	Botnet	Brute force	DoS attack	Web attack	Botnet
DARLH (proposed)	96.11	93.4	92.1	90.6	0.90	1.0	0.99	0.97
GNBH	82.4	84.3	70.4	70.1	0.32	0.43	0.65	0.54
BCH	83.1	69.2	70.6	70.45	0.55	0.65	0.76	0.42
RNSG	79.1	69.4	88.3	60.32	0.55	0.65	0.26	0.42

Abbreviations: BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; FAR, false alarm rate; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection.

TABLE 10 TNR and detection rate of Bot-IoT data set

	True negative rate				Detection rate			
Attacks Methods	Brute force	DoS attack	Web attack	Botnet	Brute force	DoS attack	Web attack	Botnet
DARLH (proposed)	96.11	93.4	92.1	90.6	0.90	1.0	0.99	0.97
GNBH	82.4	84.3	70.4	70.1	0.32	0.43	0.65	0.54
BCH	83.1	69.2	70.6	70.45	0.55	0.65	0.76	0.42
RNSG	79.1	69.4	88.3	60.32	0.55	0.65	0.26	0.42

Abbreviations: BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection; TNR, true negative rate.

6.11%, and 8.11% higher than that of the existing methods. In Web attack the FAR of the proposed DARLH 7.11%, 23.11%, and 14.11% is higher than that of the existing methods. In Botnet attack the FAR of the proposed DARLH is 16.11%, 14.11%, and 15.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

Table 10 shows the DR and TNR of Bot-IoT Data set. In this the proposed method shows high TNR and DR while comparing to other methods. The existing methods such as Game and Naïve Bayes Honey pot (GNBH), Block chain Honey pot (BCH) and RNN based Signature Generation and Detection (RNSG) respectively. In Brute-force attack the TNR of the proposed DARLH is 16.11%, 13.11%, and 9.11% higher than that of the existing methods. In DoS attack the TNR of the proposed DARLH is 8.11%, 7.11%, and 6.11% higher than that of the existing methods. In Web attack the TNR of the proposed DARLH 13.11%, 3.11%, and 6.11% is higher than that of the existing methods. In Botnet attack the TNR of the proposed DARLH is 46.11%, 13.11%, and 14.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

In Brute-force attack the DR of the proposed DARLH is 6.11%, 3.11%, and 7.11% higher than that of the existing methods. In DoS attack the DR of the proposed DARLH is 8.11%, 7.11%, and 6.11% higher than that of the existing methods. In Web attack the DR of the proposed DARLH 3.11%, 13.11%, and 15.11% is higher than that of the existing methods. In Botnet attack the DR of the proposed DARLH is 16.11%, 13.11%, and 14.11% higher than that of the existing methods, such as GNBH, BCH, and RNSG.

Figure 5 illustrates the comparison between the proposed DARLHs and existing techniques. The proposed method is compared with GNBH, BCH, and RNSG. Here, the true positive rate is taken as a measurement against a number of attacks raised in the honeypot. The comparison results show that the DARLHs are more efficient than the other existing systems. As DARLHs maintain a multilevel monitoring system (DARL and DRNN), it effectively detects DoS. The other existing systems use only standard techniques against runtime DoS dynamics. Here, the proposed DARLHs achieve 1.6% better performance compared with other existing methods. In the existing systems, RNSG is the only technique that provides a better true positive rate against the higher rate of attacks. GNBH and BCH are delivering distorted performances against dynamic honeypot traffics.

Figure 6 shows the internal and external DoS attacks. The graph represents the external attack is higher than the internal attacks. The external attack is 50% higher than the internal attacks.

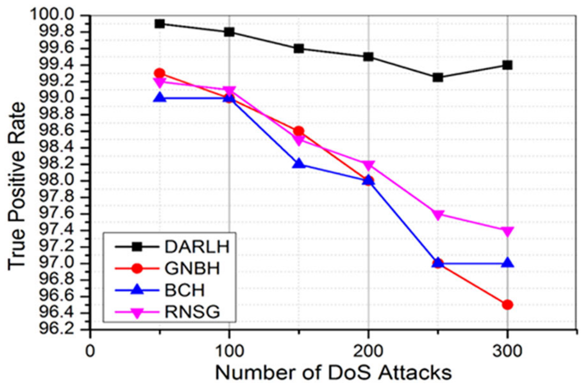

FIGURE 5 True positive rate. BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection [Color figure can be viewed at wileyonlinelibrary.com]

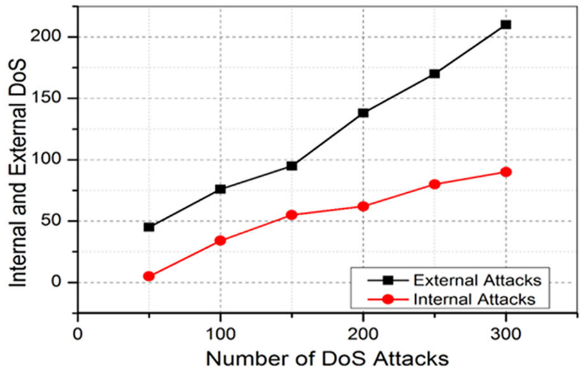

FIGURE 6 Internal and external denial of service attacks. DoS, Denial of Service [Color figure can be viewed at wileyonlinelibrary.com]

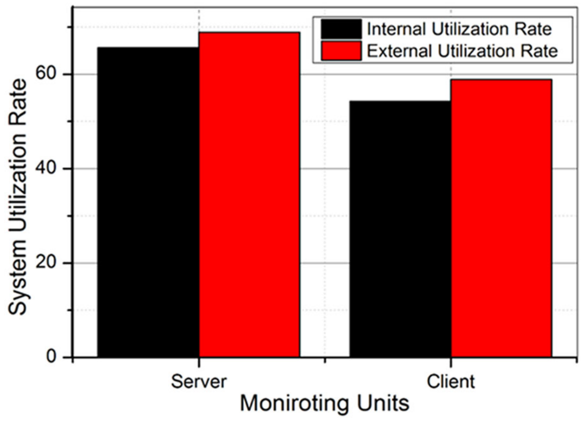

FIGURE 7 Honeypot monitoring and utilization rate [Color figure can be viewed at wileyonlinelibrary.com]

Figure 7 shows the honeypot monitoring and utilization rate of server and client. In both server and client the internal utilization rate is lower than the external utilization rate. Here, the system utilization rate of the server is higher than the system utilization rate of the client.

Figures 8 and 9 show the details of internal and external DoS detection rates. Here, the number of internal and external DoS attacks increased between 100 and 600. In both cases, the proposed DARLHs system detects the attacks effectively. Since it has individual event management queues, the events of the honeypot are efficiently distributed and monitored. In addition, both internal and external attacks are identified using appropriate authentication schemes. In existing systems, RNSG technique provides a closer performance rate in the proposed system than other systems because RNSG is a kind of DL approach. In overall performance comparisons, the proposed system produces optimal and effective results against runtime attacks in honeypot systems.[39,40]

Table 11 displays the execution time of the proposed and existing methods. On clearly observing the table, the execution time of the proposed method is low likened to the other four existing methods.

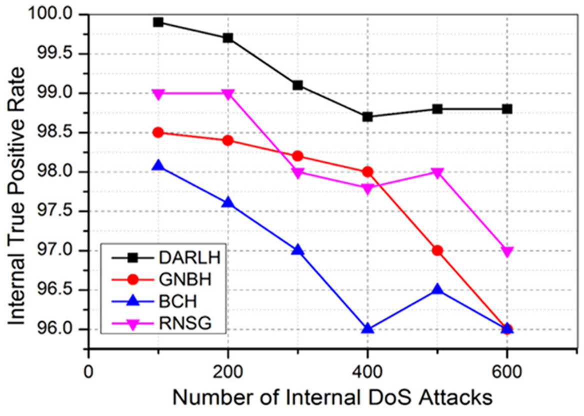

FIGURE 8 Internal DoS attack. BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection [Color figure can be viewed at wileyonlinelibrary.com]

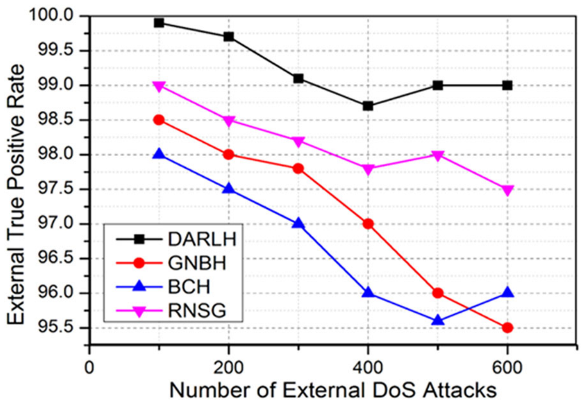

FIGURE 9 External DoS attack. BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; DoS, Denial of Service; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 11 Execution time

Methods	Execution time (s)
DARLH (proposed)	0.5
GNBH	2.7
BCH	3.9
RNSG	1.2

Abbreviations: BCH, Block Chain Honeypot; DARLH, Deep Adaptive Reinforcement Learning for Honeypot; GNBH, Game and Naïve-Bayes Honeypot; RNN, Recurrent Neural Network; RNSG, RNN-based Signature Generation and Detection.

5 | CONCLUSION

In this manuscript, the DARLHs system was proposed and implemented for detecting runtime DoS attacks on a honeypot environment. The proposed method is developed by secure event distribution techniques, server-side monitoring techniques, DARL-based Level-1 DoS detection techniques, and DARL and DRNN-based Level-2 DoS detection techniques. These techniques were implemented and compared with the existing techniques, such as GNBH, BCH, and RNSG. The comparison results demonstrate that the proposed DARLHs outperformed other techniques in all aspects against the DoS attack. These DARLHs methods provide better security against the External DoS Attack, Internal DoS attack, Brute-force attack, DoS attack, Web attack, and Botnet attack. Honeypot can be utilized as IDS that repeated some or all server activities, efficiently observer potential attackers, thereby allowing server admins to detect as well as service attacks avoid potential denial to confirm a reliably with continuous service to their proposed users. But it has the limitation in real-time applications, such as multiple attack detection strategies, corruption as well as information disclosure, to gain authority, subversion, and User Datagram Protocol (UDP) flooding. So, that an attacker can move laterally to infiltrate the real production network. To prevent this, future work is suggested.

To improve the security operations, in the future work the security operation is performed by combining honeypots with other techniques. This can be improved in the future for multiple attack detection strategies, corruption as well as information disclosure, to gain authority, subversion, and UDP flooding.

ORCID

Selvakumar Veluchamy https://orcid.org/0000-0003-1151-5352
Ruba Soundar Kathavarayan https://orcid.org/0000-0003-4082-5536

REFERENCES

1. Ghourab E, Azab M. Benign false-data injection as a moving-target defense to secure mobile wireless communications. *Ad Hoc Netw*. 2020;102:102064. doi:10.1016/j.adhoc.2019.102064
2. Kumar G, Saha R, Singh M, Rai M. Optimized packet filtering honeypot with snooping agents in intrusion detection system for WLAN. *Int J Inf Secur Privacy*. 2018;12(1):53-62. doi:10.4018/ijisp.2018010105
3. Veena K, Meena K. Implementing file and real time based intrusion detections in secure direct method using advanced honeypot. *Cluster Comput*. 2018;22(S6):13361-13368. doi:10.1007/s10586-018-1912-x
4. Mythili S, Thiyagarajah K, Rajesh P, Shajin FH. Ideal position and size selection of unified power flow controllers (UPFCs) to upgrade the dynamic stability of systems: an antlion optimiser and invasive weed optimisation algorithm. *HKIE Trans*. 2020;27(1):25-37.
5. Chu G, Apthorpe N, Feamster N. Security and privacy analyses of internet of things children's toys. *IEEE Internet Things J*. 2019;6(1):978-985. doi:10.1109/jiot.2018.2866423
6. Al-Nafjan K, Al-Hussein M, Alghamdi A, Haque M, Ahmad I. Intrusion detection using PCA based modular neural network. *Int J Mach Learn Comput*. 2012;2(5):583-587. doi:10.7763/ijmlc.2012.v2.194
7. Rajesh P, Shajin F. A multi-objective hybrid algorithm for planning electrical distribution system. *Eur J Electr Eng*. 2020;22(4-5):377-387. doi:10.18280/ejee.224-509
8. Chetna S, Swades De, eds. *Resource Allocation in Next-generation Broadband Wireless Access Networks*. IGI Global; 2017.
9. Shajin FH, Rajesh P. Trusted secure geographic routing protocol: outsider attack detection in mobile ad hoc networks by adopting trusted secure geographic routing protocol. *Int J Pervasive Comput Commun*. 2020; ahead-of-print.
10. Shrivastava RK, Ramakrishna S, Hota C. Game theory based modified Naïve-Bayes algorithm to detect DoS attacks using honeypot. In: *2019 IEEE 16th India Council International Conference (INDICON)*. IEEE; 2019:1-4.

11. Thota MK, Shajin FH, Rajesh P. Survey on software defect prediction techniques. *Int J Appl Sci Eng*. 2020;17(4):331-344.
12. Shi L, Li Y, Liu T, Liu J, Shan B, Chen H. Dynamic distributed honeypot based on blockchain. *IEEE Access*. 2019;7:72234-72246.
13. Zhou W, Shi D, Zhu W. Exponential input-to-state stability of impulsive stochastic fuzzy Cohen–Grossberg neural networks with distributed infinite transmission delays. In: *2019 Chinese Control Conference (CCC)*. IEEE; 2019:5715-5720.
14. Safarpour M, Ebrahimi F, Habibi M, Safarpour H. On the nonlinear dynamics of a multi-scale hybrid nanocomposite disk. *Eng Comput*. 2021;37(3):2369-2388.
15. Xiong L, Xu Z, Shi YQ. An integer wavelet transform based scheme for reversible data hiding in encrypted images. *Multidimens Syst Signal Process*. 2018;29(3):1191-1202.
16. Cheng P, Chen M, Stojanovic V, He S. Asynchronous fault detection filtering for piecewise homogeneous Markov jump linear systems via a dual hidden Markov model. *Mech Syst Signal Process*. 2021;151:107353.
17. Yao D, Li H, Lu R, Shi Y. Distributed sliding-mode tracking control of second-order nonlinear multiagent systems: an event-triggered approach. *IEEE Trans Cybern*. 2020;50(9):3892-3902.
18. Zhang S, Fang M, Liu H, Luan F, X, Ding Z. Reinforcement learning and adaptive optimization of a class of Markov jump systems with completely unknown dynamic information. *Neural Comput Appl*. 2020;32(18):14311-14320.
19. Zhai J, Karimi HR. Global output feedback control for a class of nonlinear systems with unknown homogeneous growth condition. *Int J Robust Nonlinear Control*. 2019;29(7):2082-2095.
20. Thu AA. Integrated intrusion detection and prevention system with honeypot on cloud computing environment. *Int J Comput Appl*. 2013;67(4):9-13. doi:10.5120/11382-6660
21. Banday M, Sheikh S. Design of secure multilingual CAPTCHA challenge. *Int J Web Portals*. 2015;7(1):1-27. doi:10.4018/ijwp.2015010101
22. Singh A, Ormazábal G, Schulzrinne H. Heterogeneous networking. *Datenschutz und Datensicherheit—DuD*. 2014;38(1):25-30. doi:10.1007/s11623-014-0007-y
23. Tsiropoulou EE, Baras JS, Papavassiliou S, Qu G. On the mitigation of interference imposed by intruders in passive RFID networks. In: *International Conference on Decision and Game Theory for Security*. Springer; 2016:62-80. doi:10.1007/978-3-319-47413-7_4
24. Vamvakas P, Tsiropoulou EE, Papavassiliou S. Exploiting prospect theory and risk-awareness to protect UAV-assisted network operation. *EURASIP J Wireless Commun Networking*. 2019;2019(1):286. doi:10.1186/s13638-019-1616-9
25. Shrivastava R, Hota C. Profile-guided code identification and hardening using return oriented programming. *J Inf Security Appl*. 2019;48:102364. doi:10.1016/j.jisa.2019.102364
26. Shi L, Li Y, Liu T, Liu J, Shan B, Chen H. Dynamic distributed honeypot based on blockchain. *IEEE Access*. 2019;7:72234-72246. doi:10.1109/access.2019.2920239
27. Kaur S, Singh M. Hybrid intrusion detection and signature generation using deep recurrent neural networks. *Neural Comput Appl*. 2019;32(12):7859-7877. doi:10.1007/s00521-019-04187-9
28. Khan R, Zhang X, Kumar R, Sharif A, Golilarz N, Alazab M. An adaptive multi-layer botnet detection technique using machine learning classifiers. *Appl Sci*. 2019;9(11):2375. doi:10.3390/app9112375
29. Srihari Rao N, Chandra Sekharaiah K, Ananda Rao A. Jananee janmabhoomischa: ICT solutions for pronational digital society. *Int J Eng Technol*. 2018;7(3.29):225. doi:10.14419/ijet.v7i3.29.18800
30. Sharma S, Kaul A. A survey on intrusion detection systems and honeypot based proactive security mechanisms in VANETs and VANET cloud. *Veh Commun*. 2018;12:138-164. doi:10.1016/j.vehcom.2018.04.005
31. Huang X, Huang Y. Mean–variance optimality for semi-Markov decision processes under first passage criteria. *Kybernetika*. 2017;53(1):59-81. doi:10.14736/kyb-2017-1-0059
32. Kotey S, Tchao E, Gadze J. On distributed denial of service current defense schemes. *Technologies (Basel)*. 2019;7(1):19. doi:10.3390/technologies7010019
33. Su L, Ye D. A cooperative detection and compensation mechanism against denial-of-service attack for cyber–physical systems. *InfSci (NY)*. 2018;444:122-134. doi:10.1016/j.ins.2018.02.066
34. Myers J. The United States patent and trademark office internet home pages. *World Patent Inf*. 1997;19:77-78. doi:10.1016/s0172-2190(97)82780-9

35. Gandhi U, Kumar P, Varatharajan R, Manogaran G, Sundarasekar R, Kadu S. HIoTPOT: surveillance on IoT devices against recent threats. *Wirel Pers Commun*. 2018;103(2):1179-1194. doi:10.1007/s11277-018-5307-3
36. KDD Cup 1999 Data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 1999, Retrieved December 15, 2018.
37. Moustafa N, Slay J. The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf Secur J: Global Perspect*. 2020;25:18-31.
38. Koroniotis N, Moustafa N, Sitnikova E, Turnbull B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. *Future Gener Comput Syst.* 2019;100: 779-796.
39. Sklavounos D. Utilization of statistical control charts for DoS network intrusion detection. *Int J Cyber-Secur Digital Forensics*. 2018;7(2):166-174. doi:10.17781/p002391
40. Silaban A, Mandala S, Jadied E. Increasing feature selection accuracy through recursive method in intrusion detection system. *Int J Inf Commun Technol (IJoICT)*. 2019;4(2):43. doi:10.21108/ijoict.2018.42.216

How to cite this article: Veluchamy S, Kathavarayan RS. Deep reinforcement learning for building honeypots against runtime DoS attack. *Int J Intell Syst*. 2022;37:3981-4007. doi:10.1002/int.22708