Chapter 1

Chapter 9: Bio/Nature-Inspired algorithms in A.I. for malicious activity detection

Andria Procopiou [1] and Nikos Komninos [1]

1.1 Introduction

Malicious software [1] is one of the main threats to networks and its assets, as well as individual users. As we approach the Internet of Things and Cyber-Physical Systems era, network traffic becomes more complex and heterogeneous. In recent years, the number of devices connected to the Internet is increased exponentially as well as big data that is produced from them. Also, each device comes with its own protocols and standards. Furthermore, computing devices operate with different protocols and standards and effective traffic monitoring becomes harder. Hence, adversaries conduct more sophisticated attacks against networks so the malicious behaviour can be more difficult to be detected. Simplistic and one-dimensional security countermeasures are likely to fail under such circumstances.

Artificial intelligence and particularly learning algorithms seems to be appropriate for detecting cyber attacks. Using machine learning, fast and accurate detection of malicious behaviour is more achievable than ever. A special branch of machine learning algorithms includes nature and bio inspired algorithms. Such algorithms followed models from nature, biology, social systems and life sciences. Some examples include genetic algorithms, swarm intelligence, artificial immune systems, evolutionary algorithms, artificial neural networks, fractal geometry, chaos theory and so on [2].

Nature/Bio-inspired algorithms have an advantage against traditional machine learning algorithms, they focus on optimisation. In detail, nature acts as a method of making something as perfect as possible or choosing the most fitted samples from a population. In practice, this family of algorithms applies these principles in the form of optimisation and finding the best solution to the problem assigned. In anomaly detection, the main objective is to identify the malicious behaviour so these algorithms use their best-fit mechanisms to detect malicious abnormalities. Another beneficial usage of nature/bio inspired algorithms is to optimise the potential features used in attacks detection. An optimal set of features is selected for efficient malware detec-

[1]Centre for Software Reliability (CSR), Department of Computer Science, City, University of London, EC1V 0HB, London, UK , { andria.procopiou.1, nikos.komninos.1}@city.ac.uk

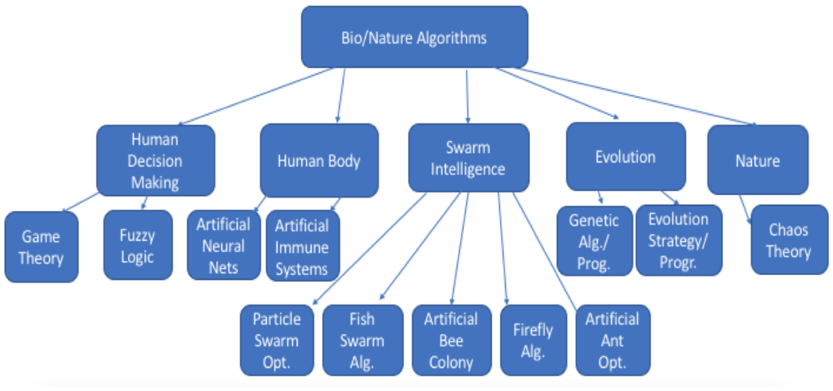

Figure 1.1 Bio/Nature-Inspred Algorithms Taxonomy

tion but also for reducing the complexity and computational burden. Additionally, nature/bio-inspired algorithms are highly flexible as they can accept a mixture of variables in terms of type and continuity. This gives us the opportunity to give a variety of different variable types which expand feature selection in such algorithms.

In this chapter, we will explore how nature/bio-inspired algorithms are applied in intrusion detection against different threats and attacks for various networks. The first section of this chapter gives an introduction and in-depth explanation of how the most popular nature/bio-inspired algorithms operate. Both the theoretical and practical concepts are explained and how these algorithms operate to detect malicious behaviour in the context of cyber security. The second section includes a selection of the most notable and complete studies of anomaly detection using nature/bio-inspired algorithms in networks and in low-resources systems such as cyber-physical systems and IoT. In the third section the techniques used and the results produced are discussed. Finally, future directions on how nature-inspired algorithms could be applied in detecting anomalies in such systems is presented.

1.2 Towards Technology through Nature

It can safely be stated that nature is the most suitable entity of solving hard and complex problems. It is able to find the most suitable solution. It can also maintain the balance between various components. Hence, computer scientists have been inspired by it, and created their own algorithms based on natural phenomena and procedures. Bio/Nature-inspired algorithms simulate the nature at solving optimisation problems. In the next section, we group and explain the most popular bio/nature-inspired algorithms as categorised in Figure 1.1.

1.2.1 Terminologies of Bio/Nature-inspired Algorithms

1.2.1.1 Populations

The term population is an important and vital concept in the field of bio/nature-inspired algorithms. In biology and nature sciences, the term population defines a group of either animals or people that belong to a specific kind/type and live in a specific place [3] . In bio/nature-inspired algorithms, the term population refers to a set of possible solutions to a specific problem. In every bio/nature-inspired algorithm the procedure starts with an initial population. An initial population is the first commencing set of solutions or candidates, before the algorithm starts functioning, with a specific size. Depending on the type of algorithm used, the population size can change.

The members of a population can be cooperative or competitive. In algorithms that consist of population members that are cooperative, no new members will be either added or deleted. On the other hand, in algorithms where the members of a population are competitive only the fittest members will be included. This process is iterative and on each iteration (called generation) the fitness of the current members is compared to the new members generated. The weakest already existing members are replaced with the strongest new members. Some old and new member generations are combined to create the most optimal solution based on rules. New members that can be potentially merged with the population must be evaluated through scoring. The score value is calculated based on the suitability of each member's solution [3].

. Also, add .(page 213, paragraph 2)

1.2.1.2 Selection

The process of selection involves choosing one or more potential solutions to the problem from the population. Depending on the bio/nature-inspired algorithm chosen, a different selection procedure is adopted [3].

1.2.1.3 Crossover

In biology, crossover is the process which a male and a female mate together to breed offspring. In bio/nature-inspired algorithms crossover does not consider genders, any two candidates can mate to produce children. Crossover is mostly used in Evolutionary Algorithms [3].

1.2.1.4 Mutation

In biology, a mutation is a change in an organism's DNA sequence. This change can be either benign or malignant. In bio/nature-inspired algorithms ”mutation” denotes the asexual reproduction, so a child could be created from a single parent. Mutation is mostly used in Evolutionary Algorithms. Through mutation, potential solutions can breed a child (new solution, that is potentially slightly better) in the next generation. If a solution has become optimal through mutation, it can become even more efficient [3].

1.2.2 Review of Bio/Nature-Inspired Algorithms

1.2.2.1 Artificial Neural Networks

Artificial Neural Networks have been inspired by the biological neural networks present in animal brains and was firstly introduced in 1943 by [4]. ANN models are inspired from biological neural networks that process an acceptable information. Just as animal brains contain biological neurons, ALRFC contain a set of interconnected nodes, called artificial neurons. Each connection between the different artificial neural networks (analogous to synapses in animal brains) helps the model of interconnected neurons (nodes) that communicate with each other by exchanging information. The neurons are connected with a set of adaptive weights that are tuned by a learning algorithm that accepts a great set of inputs and decides how the weights between the nodes are going to be adapted.

ALRFC are ideal for pattern recognition. In this simplest form, there are three types of layers in ALRFC; the input layer that accepts the pattern data; the hidden layer that applies the algorithm for deciding what the result should be; and the output layer that shows the result. Most of the models have at least an input and and output layer. The input layer accepts the pattern and the output layer gives the output pattern. The hidden layer defines the interaction between the input and the output layers. A diagram of a typical ANN architecture is shown in Figure 1.2.

ALRFC can be either supervised or unsupervised. Both of these processes are iterative. In supervised training, the similarity between the actual and the supposed output is calculated at each iteration. This is described in the form of a percentage error. The calculation of the error is called Backpropagation. At each iteration the internal weights matrices are altered towards minimising this error at a low acceptable percentage.

In unsupervised training, calculating this error percentage is not straightforward due to the model not having an expected output. Hence, the model cannot estimate the current output is from the ideal output. Instead, the model is iterated for a fixed number of rounds..

There are different types of ALRFC. Al of them have common elements such as neurons, weights, activation functions and layers. However, each type fits for

Table 1.1 ANN Architectures

Clustering	**Regression**	**Classification**	**Prediction**
Self-Org. Map(1)	Feed Forward (1)	Feedforward (1)	Recurrent (1)
-	Deep Feed-forward (1)	Deep Belief Network (1)	Deep Feedforward (2)
-	Reccurent (2)	Deep Feedforward (1)	Feedforward (2)
-	Convolutional (3)	Convolutional (1)	-
-	-	Recurrent (2)	-
-	-	-	-

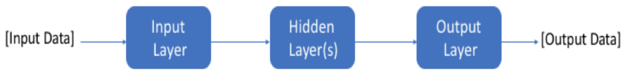

Figure 1.2   Artificial Neural Network Architecture

different tasks. Due to their versatile nature, not all of the types of ALRFC can perform equally well in every problem. There are various problems ALRFC in general can solve though, including clustering, regression, classification and prediction problems. In Table 1.1 we present a table with the most popular ANN types and the problems they can solve best, inspired by [3]. The problem is at the top and in descending order the architecture that solves it best.

ALRFC have been used extensively in Intrusion and Malware Detection. The relevant features to differentiate between normal and malicious activity are extracted from the raw data collected and fed to the input neurons of the ANN architecture. The extracted data are processed in the hidden layers of neurons and the final result is output through the output neurons.

1.2.2.2   Evolutionary Algorithms

Evolutionary Algorithms (EA) is a branch of nature/bio-inspired algorithms in Artificial Intelligence that is very popular and considered classical from all the nature/bio-inspired techniques [5]. EAs comprise of evolutionary techniques and mechanisms taken from the biological evolutions. Some examples include reproduction, mutation, selection and recombination. Since these biological models are considered nearly perfect when finding the most fitted individuals, their artificial equivalents are designed to find solutions to hard problems.

The procedure followed in different types of EAs is similar to the equivalent in nature. There is an initial population of random candidates in which natural selection occurs, otherwise called survival of the fittest. Each individual candidate is evaluated through a fitness/quality function. Then iteratively: The best candidates are chosen as parents to seed the next generation of candidates (reproduction). This is done by crossover and/or mutation applied to them. The generated candidates (offspring) are generated are evaluated by the fitness/quality function and replace the least fitting candidates of the previous generation. This process can be iterative until a sufficient solution (set of candidates) is constructed or a computational limit is met.

During this process, the fitness of candidates in consecutive populations is improved and the easiness of adaption to new environments is also improved. Below we briefly enlist and explain the most notable EAs and summarise the GA's flow in Figure 1.3.

1) Genetic Algorithm

Genetic Algorithm (GA) was firstly introduced by Holland in 1975 [6]. It is a very popular evolutionary-based algorithm especially for optimisation purposes. The algorithm functions as follows:

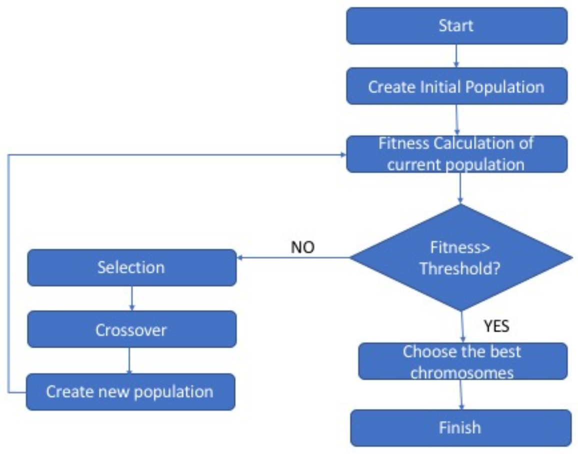
Start
Create Initial Population
Fitness Calculation of current population
Fitness> Threshold?
NO
YES
Selection
Crossover
Create new population
Choose the best chromosomes
Finish

Figure 1.3 GA Algorithm

Step 1: It begins by initialising a population of solutions (chromosomes).

Step 2: Applying a fitness function, the current chromosomes are evaluated.

Step 3: The best chromosomes are selected to generate a new generation of chromosomes through crossover and mutation.

Step 4: The new chromosomes are evaluated and exchanged with the least fit of the previous generation of chromosomes.

In Intrusion Detection, certain procedures have to be made before the training. The different groups of data (two at least with normal and attack) represent the initial different groups of chromosomes. Then the training starts with the procedure described above. During the testing/detection part, the input data is taken and an initial population's created from it. The generated population is compared to the training population. The chromosomes that the new population's chromosomes are related the closest are classified after them.

2) Genetic Programming

Genetic Programming (GP) was firstly proposed by Koza in 1992[7] and it is considered an extension of the Genetic Algorithm. GP represents the solution in the form of a tree and computer programmes are generated (instead of chromosomes). GP functions as follows:

Step 1: An initial population of computer software programmes is generated, which consists of functions and terminals.

Step 2: Each computer programme solves a problem given and a fitness value is assigned based on its accuracy.

Step 3: Once again, the best programmes are chosen to breed the next generation.

Step 4: Through mutation and crossover new computer programmes are created to form the next generation.

Step 5: Finally, the best programmes of the new generation are exchanged with the worst ones of the previous generation.

3) Evolutionary Strategy

Evolutionary Strategy (ES) methods were proposed by Bienert, Rechenberg and Schwefel in 1964 [8]. It is an optimisation algorithm which is based on the adaption of evolution theory. A special characteristic of the ES is that the mutation can be controlled depending on the type of the ES strategy chosen. Hence, both the search process and the solution are optimised, as well as the mutation parameters. Below, the most popular schemes are briefly explained [9].

(1+1) -ES : This is the scheme with the simplest selection mechanism. A real-valued vector of variables is created from the parent through mutation, by using standard deviation to each variable object. Then, newly constructed individual is evaluated and compared to its parent. The best of the two becomes the new parent for the next generation with the less fit being discarded.

(μ + λ)-ES: In this scheme, an μ number of parents is chosen from the current generation. These μ parents are responsible for creating the λ offspirng through mutation and crossover. Then the μ generation and λ offpsirng are united into one group, with only the best mi remaining, and the rest being discarded.

(μ, λ)-ES: Once again, μ parents are chosen from the current generation to generate the λ offspring (where $\lambda \geq \mu$). From the new generation only the best μ offspring individuals survive, with the parents being discarded completely.

Evolutionary Strategies are particularly effective against Intrusion Detection because they generate rules that can match the malicious traffic. To evaluate the population of suggested solutions, the dataset chosen is used to judge at which iteration the most optimal solution is obtained.

Evolutionary Programming Evolutionary Programming (EP) is again an extension of the ES methods, also using the theory of evolution [10]. It is very similar to the genetic programming method as it encompasses of computer programmes as well. Their major difference is that the structure of the programme to be optimised is fixed.

1.2.2.3 Swarm Intelligence Algorithms

Swarm Intelligence (SI) has been introduced by Beni and Wang [11] in 1989 for cellular robotic systems. The concept consists of collaborative functioning by a large number of small organisms such as bees, ants, birds and so on. Based on this concept self-organising computer network systems can operate efficiently. Under this term, the most common technique used is the Ant Colony Optimisation, which are based on the oragnisation of large ant colonies for food transporting reasons, Artificial Bee Colonies, Fish Swarm Algorithm, Intelligent Water Drops Algorithm, Firefly algorithms and so on. Below we briefly describe the common fundamental concepts all of the Swarm Intelligence are based on. A summary of how SI algorithms operate is illustrated in Figure 1.4.

Swarm Intelligence Fundamentals

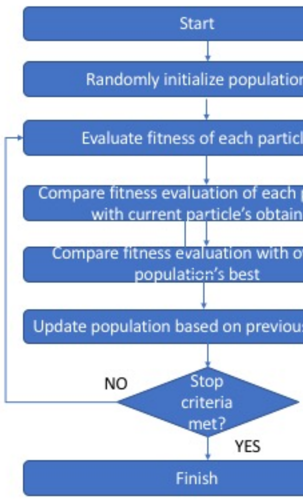
Start
Randomly initialize population
Evaluate fitness of each particle
Compare fitness evaluation of each particle with current particle's obtain
Compare fitness evaluation with overall population's best
Update population based on previous results
NO
Stop criteria met?
YES
Finish

Figure 1.4 SI Algorithm

Proximity Fundamental: The population performs a series of time and space computation calculations to function properly.

Quality Fundamental: In the presence of environment quality factors, the population shall be able to positively react.

Diverse Response Methods: In any procedures the population carries out, the means and ways shall not be limited and fixed.

Stability Fundamental: The population shall be able to remain intact and fixed in its functioning, regardless of any changes in the environment.

Adaptability Fundamental: The population must be "intelligent" enough to realise that if the computational burden can be decreased, then it shall change its behaviour.

SI algorithms can be applied in intrusion detection by constructing a set of rules for effective classification. Each organism creates a set of rules that evaluates them with the training set it is provided with. With iterations, the rules are reconstructed and extended enough to have a high accuracy percentage.

Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) was firstly introduced by Kennedy and Eberhart in 1995 [12] and has been inspired by the food searching activity of birds. It is an optimisation type of algorithm and has been extensively used due to its simplicity, computational lightness and straightforward implementation. In detail, the "particle" word of the name means the population members which are low in mass and volume can achieve a better behaviour. Every "particle" in the population can be considered a solution. Each solution consists of four vectors in the high dimensional space: current position, best position discovered yet, best position discovered yet by neighbour and velocity. Each particle rearranges its position in the search space based on two factors, its best position and its neighbour's best position during the search process until a stopping criteria is met.

Ant Colony Optimisation

Ant Colony Optimisation (ACO) was firstly introduced by Dorigo and Di Caro in 1999 [13]. It was inspired by the remarkable ability of ant species to collaboratively find the shortest path between their nest and the various food sources by making usage of pheromone trails. The more pheromone each path has, the stronger the probability is to follow it. This procedure is a type of reinforcement.

Artificial Bee Colony Algorithm

Artificial Bee Colony Algorithm (ABC) was proposed by Karaboga in 2005 [14].Inspired by the behaviour of bees, several swarm intelligence algorithms have been proposed. Mainly, there are two types, foraging and mating. One of the most popular foraging algorithms is the ABC algorithm which simulates the behavior of honeybees groups. This promotes intelligent and cooperative behavior between bees with different roles. A bee can have one of the two following behaviors; the first is when a bee finds food, the other bees are lead to the food source and the second is when bees leaving food sources for others. The bee can have three roles in these scenarios, the employed bee, the onlooker bee and the scout bee. In the ABC algorithm the place where the food is located can be a potential solution to the problem and the quality of the food maps to the fitness of the solution. The number of onlooker or employed bees corresponds to the number of the solutions to the problem. Concluding, there is only one employed bee for every food source.

Fish Swarm Algorithm

The Fish Swarm Algorithm (FSA) is a relatively new population-based/swarm algorithm, proposed in 2002 by Li et al. [15]. The FSA was based on fish schooling behaviour to search for food. A fish in the FSA algorithm is represented by a D-dimensional position and the fish food satisfaction is represented by a numerical metric. The relationship between the two fish is denoted by the euclidean distance of the two. Three basic behaviours are consisted in this algorithm, ”searching for food”, ”swarming against to a threat” and ”following towards a better result ”. *Searching for food* means that the fish randomly search for food so they can minimise the food satisfaction. *Swarming* means that the objective is to keep the food levels of fish satisfactory, keep the existing fish happy and attract new fish. *Following* means that when a fish locates food, the others will follow.

Firefly Algorithm

The Firefly Algorithm (FA) has firstly been introduced by Yang [16] in 2009 and has been inspired by the flashing behaviour of the fireflies. The algorithm consists of an iterative procedure with a population of agents (the fireflies) working together to solve an optimisation problem. The algorithm is based on the following concept of finding a solution: a better firefly glows more. Each firefly attracts other fireflies, regardless of their gender, so the searching for the optimal solution in the search space can be found more efficiently. The procedure is the following:

Step 1: All the fireflies will move towards the brightest one, regardless of their gender.

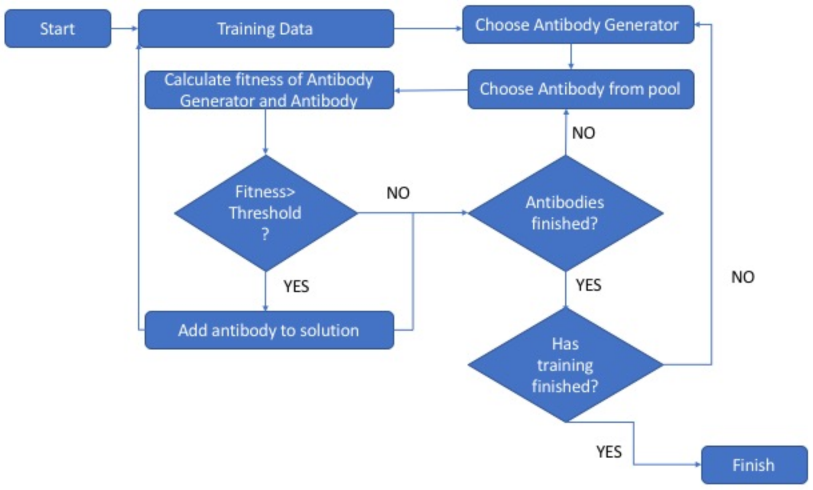

Figure 1.5   AIS Algorithm

Step 2: The more a firefly glows, the more attractive it is towards the others. The brightness can be decreased though due to distance. If there is not a firefly that stands out in terms of brightness from the rest, then they will move randomly.

Step 3: The brightness of a firefly is based on the result of a calculation of the objective function for a given problem.

1.2.2.4 Artificial Immune Systems

Artificial Immune Systems (AIS) Algorithm has been firstly introduced in 1999 [17] by Dasgupta and was inspired by the human immune system. The human immune system is a true wonder, as it is highly evolving, works in a parallel and distributive way, it is robust and can be adapted easily to any environment. The human body generates various detector cells (antibodies) to guard against non-self cells (antigens/pathogens). Antibodies are distributed to the entire body so there is no central coordinator. Each antigen is unique and independent in terms of detection. Hence, the set of body cells is mapped to the system/network/hosts to be protected and the antibodies are the IDS agents to detect the external entities (pathogens).

There are various algorithms to create antibodies to protect cells from pathogens, such the clonal selection algorithm (CSA) and the negative selection algorithm (NSA).

The CSA is based on the concept of acquired immunity where in a specific way B and T lymphocytes respond in a better way to antigens over time. The negative selection algorithm is inspired by the biological concept of identifying and deleting self-reacting cells that attack self-tissues.

NSA operates as follows: normal data are defined as self-patterns. Then, a great number of random patterns is generated and compared to the self-patterns. If the new generated pattern matches one of the self-pattern, then it becomes a detector otherwise it is discarded. Next in the monitoring state, if there is a match between a detector and an incoming traffic part, then an anomaly is present. Neighborhood Negative Selection (LRFC) on the other hand operates in two stages. In the first stage, feature extraction takes place and then these features are used to train the algorithm. The system extracts the duration/direction of the flow and the number of packets/bytes. In the second stage, the LRFC algorithm is used for detection. A quantitative approach is adopted to detect instead of a single threshold. A less used algorithm of AIS is the Dendritic Cell Algorithm (DCA). DCA is based on how dendritic cells operate in natural immune systems as activators.

AIS relies on mutation to make its decisions and operates in the following way:

Step 1: *Initialisation:* The potential solutions to the problem (antibodies) are initialised. Antigens mark the value of the objective function that must be optimised.

Step 2: *Cloning:* The antibodies are evaluated on how fit they are. The best antibodies are cloned proportionally, with the best being cloned the most.

Step 3: *Hypermutation:* In this stage, the best antibody's clones are mostly mutated and the worst antibody's clones are lessly mutated. Next, the clones are evaluated with their original antibodies. The best antibodies are kept for the next generation, with the best being discarded, for the next generation. This mutation can be gaussian, exponential or uniform.

In Intrusion Detection , AIS consists of three phases. Firstly, the algorithm defines itself by learning the normal behaviour so it can construct a baseline. Secondly, it generates detectors thrgh he T-cells prcess. Thirdly, the input data are compared to all the T-detectors. If there is a match then there is no intrusion.

1.2.2.5 Fuzzy Logic

Fuzzy Logic (FL) was proposed in 1965 [18] by Zadeh. FL concept mimics the human brain interpretation of uncertain information. Similar to the brain operation, FL replies to inputs with approximate (truthy/faulty) rather than distinct and exact reasoning. In reasoning some questions have various answers, based on a set of knowledge and rules that form a specific solution to the problem. Likewise, each FL reply has a degree of truth and in combination with probabilities they form a fuzzy set.

1.2.2.6 Chaos Theory

Chaos is a sub-area of mathematics that was firstly formed by Edward Lorenz in 1972 [19]. In contrast to most of the scientific models that are responsible for modelling predictive behaviour such as gravity, electricity, chemical reactions and so on, Chaos Theory can model non-linear behaviour that has no means of being predicted such as the weather, stock market and so on. Formally, in the context of complex systems there are always subtle patterns, iterative feedbacks along with iteration itself, underlying similarities in behaviour and overall connections between the various states of a system. A very important aspect in chaos theory is that in a complex system there is the so called ”sensitive dependence on initial conditions”, the ”butterfly effect”. In detail, this effect denotes that a small change in a single state of a complex deterministic non-linear system can cause major changes in a later state/outcome of it.

In the context of Intrusion Detection, the IDS system closely monitors the traffic and through calculations can understand at what extend it starts behaving chaotically. If there is an indication of chaos, then there is likely an intrusion.

1.2.2.7 Game Theory

Game Theory (GT) has been applied in many areas including mathematics, psychology, economics, computer science, political sciences, and gambling. GT is inspired from mathematical modeling and decision-making. Although multiple and different approaches to Game Theory have been proposed it has been firstly introduced by Neumann in 1928 [20]. It studies cooperation and conflict between individuals who make decisions upon particular rule-scenarios. The theory consists of games played between players where games can be cooperative or non-cooperative depending on the players intentions. Games are also based on self-interest or common interest.

In the context of Intrusion Detection, a game is played between the IDS and the possible attacker. Based on the model constructed and the assumptions considered, a decision is made on whether there is an intrusion.

1.3 Cyber Attacks and Malware Detection

In cyber space there are numerous cyber attacks that can be conducted against networks and/or individual assets. These attacks can threaten one or more of the main Cyber Security Principles : Confidentiality, Integrity and Availability. Below we give a brief explanation of the most popular cyber attacks & malware and their impact on a network and/or asset, as well as which Cyber Security Principle they mostly negatively affect.

1.3.1 Distributed / Denial of Service Attacks

A Distributed / Denial of Service Attacks (D)DoS attack can be volumetric or vulnerability exploitation-based or reflection-based. A volumetric DoS attack consists of an attacker sending a massive volume of requests in an attempt to flood the target machine and make it unable to accept legitimate requests from normal users [21]. In a vulnerability-exploitation (D)DoS attack an attacker is taking advantage of a system or protocol or communication weakness to make the target machine unavailable to legitimate requests from normal users. In a reflection (D)DoS attack the target is responding with the received challenge. A (D)DoS attack can also occur in any of the Transmission Control Protocol/Internet Protocol (TCP/IP) Layers. (D)DoS attacks mainly threat the Availability Security Principle.

(D)DoS attacks can occur in all the Open Systems Interconnection (OSI) layers. Below we briefly describe the most popular (D)DoS attacks across all layers.

Physical Layer: In the Physical Layer we have Jamming attacks.

A *Jamming* attack consists of an adversary flooding the physical medium with signals in order for legitimate packets not being able to be transmitted normally.

A Jamming attack has multiple forms. The most popular is the Constant Jamming (D)DoS attack where a burst amount of radio signals are constantly sent. The second type is the Deceptive Jamming (D)DoS which falsely sends radio signals in an attempt to make the target unavailable.

Link (MAC) Layer:

In the Media Access Control (MAC) layer we have the *ARP Poisoning* attack where the attacker performs MAC address spoofing so it can send Address Resolution Protocol (ARP) Messages to a network. Hence, messages are redirected to him instead of the normal user. Therefore, the attacker can cause a (D)DoS attack by dropping the packets.

In wireless networks, a node that wants to connect to a wireless network, scans the environment to connect to an available network by sending Probe Requests. The access point responds back, providing information about the network. In a **Probe Request Flooding** attack, the adversary sends a burst amount of probe requests to the access point to make it unavailable to legitimate requests.

Another popular attack in the MAC layer consists of the *Authentication Request Flood*.The attacker who has already perfromed a MAC spoofing attack to try and authenticate themsevles to the access point by sending authentication requests. The attacker aims to flood the access point with authentication requests.

Network/Transport Layer: In the network and transport layer we have a variety of different types of (D)DoS attacks.

In a *SYN-Flood (TCP-Flood) (D)DoS* the attacker sends a great number of SYN packets to the target until the target becomes unresponsive to legitimate requests. In a normal situation a clients initiates a TCP connection by sending a SYN message to the server. The server responds with a SYN-ACK to the client, the client responds back with an ACK message and the connection is established (three-way handshake).

In a *UDP Flooding (D)DoS* the attacker sends a large number of User Datagram Protocol (UDP) packets to random ports of the target.

In a *Gray Hole (D)DoS* the attacker selectively drops packets that are to reach a particular destination.

A *Black Hole (D)DoS* is similar to the gray-hole attack but the attacker drops all packets passed by it.

An *ICMP Flooding (D)DoS* , otherwise called Ping Flooding (D)DoS attack, sends a burst amount of Internet Control Message Protocol (ICMP) echo request packets to the server.

In a *Low-Rate (D)DoS* the attacker exploits TCPs retransmission timeout mechanism and sends low-rate traffic to the target so the connection can be kept alive. As time goes by, the attacker opens more and more connections and eventually makes the target unavailable to legitimate incoming connections.

In a *Ping of Death (PoD) (D)DoS* the attacker aims to make the target unavailable by sending either malformed or oversized packets using the ping command.

In a *Teardrop (D)DoS* the attacker sends mingled IP fragments with either overlapping or over-sized payloads to the target server in an attempt to make it unavailable.

In a *Land (D)DoS* the attacker sends a TCP packet, which contains the targets IP Address as both the source and the destination. This causes the victim to send replies to itself over and over again, resulting into it becoming unavailable.

The *Fraggle (D)DoS* is similar to Smurf attack but instead of ICMP echo packets the attacker sends UDP echo packets to the target.

The *Smurf (D)DoS* attack can either be a simple or distributed DoS attack where the attacker(s) through IP Spoofing the victims IP address sends a large number of ICMP echo requests to the IP broadcast address. As a result, the victims computer is becoming extremely slow and eventually becoming unavailable. Finally in the application layer we have a fair number of different (D)DoS attacks.

The *SIP Flooding (D)DoS* attack is conducted in the Session Initiation Protocol (SIP), which is an application layer protocol used in Internet telephony for both voice and video calls as well as instant messaging. The attack floods the target with either valid or invalid calls or messages.

In an *HTTP Flooding (D)DoS* the attacker sends a burst amount of legitimate get or post request to the victim resulting into the target becoming unavailable to legitimate requests in the Hypertext Transfer Protocol (HTTP). Since the requests are perfectly legitimate the server has no ways of rejecting them.

In a *DNS (D)DoS* the attacker sends Domain Name System (DNS) queries that at first are small in size but then become large in size. The attacker redirects these messages to the victims IP address resulting into a targets unavailable state.

In a *Slow Rate (Request and Post requests and Response) (D)DoS* the attacker takes advantage of the servers waiting time for a get/post request to be completed (time-out time). The server is waiting for a specific amount of time for the client to send the request before closing the connection. The attacker instead sends a request very slowly, tricking the server into believing that simply the request is coming from a slow connection. In time, the attacker opens more and more connections to send requests that are never completed. As a result, the server becomes unavailable. In response attacks the attacker reads the response sent to him very slowly by having a much smaller window-size than the servers send buffer size.

In *SSL (D)DoS* attacks the adversary exploits the Secure Socket Layer (SSL) hanshake authnetication mechanism. In detail, the attacker sends bogus data to the target or exploits the functions to the SSL encryption key negotiation procedure.

1.3.2 Botnets

A botnet is a set of devices/machines that are connected to the Internet which, at some point, have been infected by a malware and are controlled by an adversary, the botmaster. A botnet can perform a (D)DoS attack, send spam mail or engage in click frauds [22]. This can result into the network/target not functioning fully thus, the Availability principle is threatened.

1.3.3 Malware

Malware, an acronym for Malicious Software is a general term encompassing any malicious or intrusive programme. It can take the form of a an executable code, script, active content and so on. A malware exploits any known or unknown vulnerabilities of a system, network, machine, protocol and so on so it can illegally execute malicious code to steal data, destroy assets or abuse services and/or functions [22]. The most popular types of malware are briefly explained below.

1.3.3.1 Viruses

A virus is a malicious software that once installed, through user interaction, it starts making copies of itself and infects various types of files and/or programmes. Through it the Confidentiality and Integrity principles are affected [22].

1.3.3.2 (Remote Access)Trojan Horses

A Trojan Horse is a type of malware that masquerades itself as a legitimate, useful programme in order for the victim to be tricked and install it. It has taken its name from the Ancient Greek Trojan Horse used to stealthily invade the city of Troy. In contrast to viruses and worms, Trojans do not usually inject or propagate themselves in other files. A Remote Access Trojan is a special type of Trojan that gives to the adversary remote access to the victim's computer. Through it the Confidentiality and Integrity principles are affected [22].

1.3.3.3 Rootkits

A rootkit is a special type of malware that its main strength is that it remains undetected. To achieve that, it modifies the victim's operating system so it remains hidden from the user. In detail, the rootkit is able to hide malicious processed from being visible in the list of active processes. It also eliminates any access to its files. Through it the Confidentiality and Integrity principles are affected [22].

1.3.3.4 Backdoors

A backdoor gives the ability to the adversary to invisibly access and manipulate remotely a machine by bypassing any form of authentication. Through it the Confidentiality and Integrity principles are affected [22].

1.3.3.5 Spyware

A spyware is a malicious software that once installed on a victim's comptuer, starts illegally gathering information about assets without the user's consent. Through it the Confidentiality and Integrity principles are affected [22].

1.3.3.6 Worms

A worm is a type of malware that, unlike viruses, can replicate itself without any user interaction and spread to other computer when connected to a network. Through it the Confidentiality and Integrity principles are affected [22].

1.3.3.7 Ransomware

A ransomware is a special type of malware that either aims to encrypt a victim's entire data and/or programmes or block access to the entire machine. To reverse these malicious actions, a ransom must be paid to the adversary [22].Through it the

Confidentiality and Integrity principles are affected as well as Availability since the machine is unavailable to the user.

1.3.4 Probe Attacks

Probe attacks are scanning procedures an attacker follows to collect useful about potential vulnerabilities a machine or an entire network has. Based on the gathered information the attacker can attempt to exploit the victim at a later stage [22].Through it the Confidentiality principle is affected.

1.3.5 Buffer Overflow

Sometimes, due to poor programming code, applications and software have a serious vulnerability. While data is being written to the buffer during a process, the buffer's boundary is overrun and nearby memory locations are overwritten. An attacker can exploit this vulnerability through malformed inputs to gain unauthorised access. Through it the Confidentiality and Integrity principles are affected [22].

1.3.6 Brute Force Attack

A brute force attack consists of an attacker attempting to break a password by trying a great amount of random combinations to get it right [54]. Here, the Confidentiality principle is affected [22].

1.3.7 Masquerading Attacks

In masquerading attacks, an adversary manages to steal the identity of a legitimate user and masquerades themselves as normal entities in an attempt to exploit a system or a network [22].

1.3.8 Datasets used in Intrusion Detection

1.3.8.1 DARPA Dataset

The MIT Lincoln Laboratory along with the Defence Advanced Research Projects Agency (DARPA ITO) and Air Force Research Laboratory (AFRL/SNHS) have collaborated together to create an Intrusion Detection dataset from 1998-1999. The DARPA 1998 dataset consists of seven weeks with normal and attack traffic. The dataset set consists of two weeks of attack-free traffic, with one week of new attacks along with the old ones from the DARPA 1998 dataset [23].

1.3.8.2 KDD-99 Dataset

The KDD-99 Dataset became public in 1999 and has been one of the most popular IDS datasets used in literature. It has been based on the captured data of the DARPA 1998 dataset. It contais seven weeks of raw network traffic with an approximate total of 4,900,000 connections. A total of 41 features have been suggested and were used to differentiate between normal and malicious traffic. There are various types of cyber attacks present in the dataset. These attacks are categorised into four groups, Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L) and Probing Attacks. DoS and Probing attacks have already been explained above. U2R denotes a set of attacks in which the attacker manages to gain root access. R2L denotes a set of attacks in which the attacker exploits a certain vulnerability to gain access to a local machine in the network[24]. A newer version of the KDD99 dataset, the NSL-KDD dataset, attempts to improve the KDD-99 dataset's issues highlighted in the literature's studies[25].

1.3.8.3 ISCX IDS 2012 Dataset

In the ISCX IDS 2012 Dataset, various multi-stage attacks scenarios were conducted such as SSH Brute Force attacks, (D)DoS attacks and through IRC Botnets and internal Infiltrating of the network[26].

1.3.8.4 ISCX IDS 2017 Dataset

In the ISCX IDS 2017 Dataset, multi-stage attacks scenarios were conducted with more diversity. Some examples were various (D)DoS ADoS slowloris , DoS Slowhttptest, DoS Hulk, DoS LOIT and DoS GoldenEye as well as the botnet ARES. Brute Force attacks were also conducted such as SSH and Brute Force attacks. Also, web-based attacks were conducted such as web Brute Force ,XSS and SQL Injection. Finally, port-scan maliucious behaviour was simulated [27].

1.3.8.5 Botnet Dataset

In the Botnet Dataset, various botnets were used to attack the network such as Neris, Rbot, Virut, NSIS,IRCBot , Menti, Sogou, Zeus, Weasel, SmokeBot, Murlo [28].

1.3.8.6 CIC DoS dataset

In the Canadian Institute for Cybersecurity (CIC) DoS dataset, the attack behaviour was focused on Application Layer (D)DoS attacks. The attacks conducted were high-volume as well as low-volume. The High-volume attacks were HTTP GETs, DNS queries, SIP INVITEs. The Low-volume attacks were low-rate attacks, Apache Range Header attack, Slow-Rate DoS Slowhttptest, DoS Hulk, DoS LOIT and DoS GoldenEye [29].

1.3.8.7 The AWID dataset

The AWID dataset has been constructed in 2015 and consists of a dataset with normal and attack traffic targetting 802.11 networks [30].

1.3.8.8 The UNSW-NB15 dataset

The UNSW-NB 15 dataset was constructed by the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) consisting both normal and attack data. This data set has nine types of attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms [31].

1.4 Bio/Nature-Inspired Algorithms Studies in Intrusion Detection

1.4.1 Game Theoretic Studies

The authors in [32] used GT with Nash Equilibrium. They designed a cooperative IDS in an attempt to increase accuracy and to detect new threats. The authors proposed a trust management framework so the IDSs could be connected in a trustful way, reduce the impact for compromised IDSs and reduce the false positive rates and be scalable. When an IDS evaluates traffic to detect potential DoS attacks but has limited knowledge it seeks help from other IDS by sending requests to gain extra feedback from them. There is a central component that organizes the communication between the different IDSs. The authors evaluated their solution against DoS attack so when a malicious node is detected it is removed from the network.

The authors in [33] also used Game Theory to prevent and detect (D)DoS attack in Ad Hoc Networks. The game theory concept was applied as a form of multiplayer game in combination with cryptographic puzzles. In an ad-hoc network, when a node wishes to send a request it needs to solve a puzzle first and then become a member in a group. The nodes belong to each group play a multiplayer game that is divided into 2-player sub-games. The winner of each group, (highest score) is served by the requested node. The games are continued until all requests are served. In the end, there will be one player left at each group. In that case, that game is played between the last node and the node responsible for authorizing the requests. By doing that the authors ensure that the bulk of the computation overhead is handled by the requesting node which checks the winning results. In case of a malicious node is already present in the network it will need to process a high computation puzzle. The authors found that the optimal number of players for each group would be between 2 and 20. The solution also is efficient for the ad hoc nodes of the network.

1.4.2 Evolution Strategies Studies

The authors in [34] have suggested an online intrusion detection framework able to detect SIP Flooding attacks using various Evolutionary Algorithms, including Evolving Radial Basis Function, Fuzzy Ada Boost, Genetic Classifier, Extended Classifier, Supervised Classifier and Continuous Ant Miner Classifier. Their framework consists of six components. The packet sniffer captures the SIP Traffic and stores it in a buffer, which can hold up to 500 packets. As soon as this buffer is full the feature extractor component begins analysing the packets and extracts values for a set of features selected. These features are invite, register, bye, ack, options, cancel, update, refer, subscribe, notify, message, info and prack requests. The values of the features are normalized according to the total number of SIP request messages. The features also include response messages, such as the number of Success, Redirection, Client Error, Server Error, Global Error that are normalized according to the total number of SIP messages. With the help of rules generated from the Evolutionary Algorithms it is decided if there is an attack. Supervised Classifier and Extended Classifier have been proved to be effective with 82.17% and 77.00% against chunk flood attacks and harmonic flood attacks..

The authors in [35] use ES to generate rules to match anomalous connections from the DARPA Dataset, including DoS and (D)DoS attacks. As they have stated in their work, the majority of research is focused on GA approaches so they are the first to use ES to generate rules for detecting anomalous connections. The attributes the authors have used for detecting anomalies are the source IP address, the destination IP address, the source port, the destination port, the duration of the connection, the state, the protocol, the number of bytes sent by the source and the number of bytes sent by the destination nodes. The range of values for each attribute have a predefined set of values and use ES to find the optimal range. The authors do not present experimental results, although their suggestion is important for research.

1.4.3 Genetic Algorithms Studies

There are numerous studies that make usage of genetic algorithms to detect various cyber attacks. GA is an excellent algorithm for optimising the features used in intrusion detection, managing to not only reduce the overall complexity of the system but also achieve a better detection rate against various cyber attacks.

The authors in [36] designed an IDS for MANETs using GA to detect black hole and dropping routing DoS attacks. They defined a set of characteristics for the GA algorithm to consider (whether connections was from/to similar port/host, number of wrong fragments within the connection, connection SYN errors, percentage of connections to similar/desimilar services and hosts and so on). Using a trained set they constructed the population. Then, by introducing the test data to the GA and by performing analysis (selection, crossover, mutation), test data were categorized. The authors report a detection rate of 95%.

In [37], the authors have designed a multi-layer approach intrusion detection system by using the genetic algorithm. They followed a multi-layer IDS to detect attacks from the KDD99 Dataset. Each layer operate as a filter for each group of attacks (there were four layer, one for each group of attacks) and blocks any malicious activity so there is no need for further investigation. The authors report that as the number of rounds of the GA is increased so does the accuracy of the results but the complexity as well. They report an overall of 90% of potential accuracy but the time complexity is reported to be relatively high.

The authors in [38], have also made usage of GA to detect the same family of attacks from the KDD99 Dataset in wireless networks. They have compared their results with other studies, which consist of studies using ALRFC, and their solution is reported to have a detection rate 95.67% training accuracy and 97.57% testing accuracy. They have also made usage of only 16 features to achieve these results from a total of 41 features.

1.4.4 Fuzzy Logic Studies

In intrusion detection, FL is often used in combination with other bio-inspired algorithmic techniques such as GA. The authors in [39] combined GA with FL to detect KDD99 set of attacks, including DoS attacks. For the detection using the GA algorithm they have used a set of characteristics; source IP address, destination IP address, connection duration, protocol type, source port, destination port, and destination host service. With FL specific rules, the detection rate is increased and the error rate is decreased when the input is ambiguous. The results indicate that solution being more accurate, faster and lighter in computational resources than GA and FL when used separately. Specifically, the detection rate is high (>95%), the execution time is 12,500 milliseconds and the memory allocation in less than 25 MBs.

In [40] the authors have combined the GA along with FL to create the Fuzzy Genetic Algorithm (FGA) for their research. The algorithm was detecting all the attacks from the KDD99 Dataset . The algorithm commences by randomly choosing rules and then using the evolutionary algorithm to improve the rules during the training phase. Then, during the testing phase the final rules are used for classification of the testing data. The new algorithm was trained with traffic collected by the researchers themselves and can also detect unknown attacks. Through a series of experiments the FGA has detection rate percentage above 95% and false positive rate less than 1%.

GA was also used as a feature selection technique before proceeding to the detection part. In [41] the authors have used GA to choose the most relevant and accurate for detection features from the KDD99 dataset. In that way, the authors aimed to perform a faster, lighter and more accurate detection. For the detection part, a Support Vector Machine (SVM)-fuzzy based algorithm was used for detection. Authors' report claims their results are better when using their solution rather than SVM by itself or the SVM-fuzzy algorithm without the GA performing the feature selection. Specifically for the detection of DoS attacks, the detection rate is reported to be 98.3% and the error rate to be 2.7%.

The authors in [42], have used fuzzy logic to detect (D)DoS jamming attacks at the Physical and MAC layers in IEEE 802.15.4 low rate wireless personal area networks. They have taken into consideration bad packet and signal-to-noise ratios to detect such attacks. Bad Packet ratio is measured at the receiver side and consists of the total number of bad packets received by a node, divided by the total number of all packets received by the node, over a specific amount of time. Signal-to-Noise ratio consists of the ratio of the received signal power at a node to the received noise at the node. A collection of objects has a fuzzy set which maps each object of the collection to a membership value (by applying a membership function to it) from 0 to 1. In this case two set of objects exist, one for the signal-to-noise ratio and one for bad packet ratio. Through fuzzification, each of the two values produced from the two metrics fall among low, medium and high. According to certain rules one final result is produced that is defuzzied and a final decision is made on whether there is jamming attack or not. Throughout simulating different types of jamming attacks the detection rate has an average of 99.75% and a false positive rate of 0.01%

1.4.5 Swarm Intelligence Studies

The authors in [43] used the Ant-Colony-Optimisation (ACO) to detect UDP DoS attacks on port 7, specifically Fraggle and Teardrop attacks. The agents (ants) iteratively monitor the network activity flow. More specifically, each ant represents an IDS agent, which is responsible for monitoring the network activity. Each agent/ant moves from a state/step *a* to a state/step *b*, where each state represents network pattern knowledge. On each state, depending on the incoming packets received, a set of feasible expansions to the ant colony is calculated and the agent moves to the highest probability expansion. The probability for each move is calculated based on step attractiveness and step trail. Trails values are automatically updated when all agents/ants complete their state transiting. An increase or decrease on trail levels indicates a good move/legitimate behavior or a bad move/malicious behavior. The results indicate that the detection rate is reaching 80%.

The authors in [44] used the Ant-Colony-Optimisation algorithm and focused on detecting low-rate (D)DoS attacks. The system firstly collects information from incoming traffic such as traffic flow, packet arrival time, packet attack rate and arrival time interval by manual setting. Next, the multi-agents installed in the network are working cooperatively towards the detections of malicious activity and tracking down the IP addresses of the attackers in a cooperative way. To change state, the traffic flow and time density from router to router is calculated. The proposed technique can also be used to detect attacks in neighboring paths. The average detection rate is 89%.

The authors in [45] designed an Intrusion Detection System to detect attacks of the KDD99 Dataset, including various types of (D)DoS attacks using the Ant Colony Optimisation algorithm. To detect anomalies in the traffic they used IP Protocol segments, the total length of each packet, the source port and destination port in the TCP and UDP. They report that their algorithm has a precision of 96.94%.

In [46] the authors have used theArtificial Bee Colony algorithm to detect various MANET-specific attacks such as Flooding attacks, Balckhole attacks and Wormhole attacks. The authors extend the ABC algorithm to dynamically update the generated profile as MANET networks are not static (due to the nodes mobility). Their system has three stages, training, detection and update states. Each of the nodes in the MANET create a set of spherical detectors. The ABC algorithm applies Monte Carlo estimation to prevent the overcreation of detectors and Gaussian Local Search to refine the detectors generated. The spherical detectors are responsible for differnetiating malicious from normal behaviour. The spherical detectors at regular time intervals are partially updated. Monte Carlo estimates when the detectors shall be totally updated. In their results, they report a Detection Rate 96.11% with FPR 1.45%.

The authors in [47] combined PSO with Fast Learning Networks (FLN) which is a specific type of ANN to detect malicious activity in the KDD99 dataset. FLN comes with the disadvantage of not consisting of optimal weights. Hence, its accuracy is decreased. For that reason, the auhors use PSO to select optimal weights for the neurons. PSO creates particles, each of them representing one candidate solution for the weights of the FLN. The authors have compared their solution to other types of ALRFC such as Evolutionary Learning Models and showing that their solution has supprassed related solutions. Furthermore, they report he accuracy has increased for all models with increasing the number of hidden neurons. In detail, they report a TP rate for Normal Traffic of 99.7%, for (D)DoS attacks 98.11%, for

User to Root Attacks 80.40%, for Remote To Local Attacks 54.55% and for Probe Attacks 80.01%.

1.4.6 Artificial Neural Network Studies

In most of the bio/nature-inspired studies for intrusion detection, ALRFC are used in combination with other bio/nature-inspired algorithms that their main objective is to optimise the features. Some notable studies are briefly described below.

The authors in [48] used a combination of SI and ANN to detect malicious activity in the KDD99 dataset. In detail, they have made usage of the ABC algorithm with a BP ANN in an attempt to construct an effective neural network architecture to avoid overfitting problems. The authors justify their choice through the following argument: the ABC algorithm's objective is to optimise the BP neural network's training process(construction of network weights and thresholds, initial training values) until it reaches an acceptable accuracy. For evaluating the model they have used 10% of the KDD dataset. They report a squared error of approximately less than 0.25, while a traditional BP neural network is higher.

In [49], the authors have combined the GA along with ANN, attempting to increase the detection rate. The have used the Center for Applied Internet Data Analysis (CAIDA) dataset. The GA was used for feature selection to find the most important features from a total of 43 and exclude all the irrelevant or redundant ones. The ANN was used for detection by taking the most relevant features as inputs, using the MLP method for defining the architecture. The proposed solution had a detection rate of 99.997% and a False Positive Rate of 0.002%.

In [50], the authors designed a Neural Network IDS to detect DoS attacks with the KDD99 Dataset. They used the LVQ algorithm to structure the architecture of the neural network. They conducted the experiment 10 times using both BP and LVQ algorithms to compare results. The ANN using the LVQ algorithm achieved an average DR of 99.723% and a 0.277% FPR, while the BP-ANN achieved an average DR 89.9259% and a 0% FPR.

The authors in [51] have used ANNs and the MLP method to detect DoS attacks in the NSL-KDD dataset (11 in total). They have constructed 11 different MPL-Neural Networks one responsible for detecting each attack. They have altered the features that are normally used to train the neural network differently. Specifically, they changed the connection feature. If the ”connection” is legitimate the feature value is set to 0. If it is malicious it is set to 1-39, depending on the attack it corresponds to. For the flag feature instead of Boolean value they have defined 11 flags, each corresponding to a different state of the connection. Lastly, service now indicates 64 different kinds of services. In their results they report an average detection rate of 96.6% and a false positive rate of 3.4%

The authors in [52] have used MLP Artificial Neural Networks to detect (D)DoS attacks in the AMI of the Smart Grid. Their proposed method consists of three phases, the training, the feature extraction and traffic filtering phases. The features chosen are the packet headers, source and destination port, windows size and flags. The authors were particularly interested to evaluate their proposed method under the possibility of accepting impure traffic. Impure traffic consists of malicious traffic mixed with normal traffic. They have tested their technique under five impure datasets they have created. They report the best results to have a false acceptance rate of less than 8% and the false rejection rate to be less than 5%.

The authors in [53] have used Artificial Neural Networks to detect (D)DoS attacks in 802.16 (WiMax) networks, which is used in the Smart Grid, The proposed system consists of three stages, feature extraction, process of the features and features forward to the Neural Network for classification. The features extracted are the hurst parameter incoming traffic, the average bit rate of received messages, the increment bit rate of the received message, the entropy and conditional entropy of received MAC address and the amount of mutual information that use entropy and conditional entropy of the received packet. For the maximum number of subseries size (=256) the maximum true positive rate is 85% and the minimum false positive rate is 15%.

1.4.7 Artificial Immune Systems Studies

The authors in [54] used a variation of the NSA, the Neighborhood Negative Selection (LRFC) algorithm to detect DoS flooding attacks focusing on the ICMP, TCP and UDP protocols, from the DARPA dataset. NSA operates as follows: Normal data are defined as self-patterns. Then a great number of random patterns is generated and compared to the self-patterns. If the new generated pattern matches one of the self-pattern then it becomes a detector, otherwise it is discarded. Then, in the monitoring state if there is a match between a detector and an incoming traffic part then an anomaly is present. LRFC has two stages. The system extracts from traffic protocol type, duration of the flow, number of packets/bytes from outside to inside and vice versa. In the first stage, feature extraction takes place and then the set is used to train the algorithm. In the second stage, the LRFC algorithm is used for detection. Neighborhoods represent data, so therefore no single threshold is used to detect anomalies but rather a quantitative approach is adopted. The results indicate that the proposed system can detect the attacks in an efficient way. In detail, the TN rate has not fallen below 8.5% and the maximum value for FN rate was no more than 15%.

The authors in [55] apply the CSA (AIS) algorithms. They proposed an IDS for the Smart Grid infrastructure. The attacks considered were from the KDD99 dataset and the DoS group of attacks were included. The false positive rate (FPR) is only 0.7% while the false negative rate (FNR) is 21.02% for the clonal selection algorithm and 1.3% and 26.32% for the AIS, respectively. Their high FNR value is because of the U2R attacks so it is assumed that the proposed solution is effective against DoS attacks. U2R attack (User-to(2)-Root) is when the attacker initiates the attack as a normal user and by exploiting the systems vulnerabilities attempts to gain root privileges.

The authors in [56] focused in detecting SYN-Flooding attacks using DCA. DCA correlates the different data-streams as signals and antigens. The signals represent the behavior of the system while being categorised. The output signal value changes according to the input signals received. If that value changes in an abnormal way it is removed from the population for analysis. Two series of experiments were conducted, one for selecting the best attributes to pass them to the algorithm and another for the actual detection. Through a series of experiments the authors claim perfect detection rate and only 0.17 FPR.

The authors in [57] have used AIS generation algorithms in the NSL-KDD dataset. In particular, they have used Negative Selection Algorithm (NSA) and the Clonal Selection Algorithm (CSA) with different set of features at each experiment. For the NSL-KDD dataset, the authors found the following thresholds showed the most promising results: 39 features used a threshold of 20, 22 features used a threshold of 9, and 13 features used a threshold of 6. For 39 features the DR is 86.86 , for 22 features the DR is 77.23 and for 13 21.56.

In [58] the authors have used AIS to design a distributed IDS to detect cyber attacks in the KDD99. Two types of antibodies are constructed (through positive and negative selection), one for malicious and one for normal traffic. In practice, antibodies are formed as hyper-shpere shapes with a specific centre and radius. The centre is denoted as a vector of the values for the features chosen. Hence, Euclidean distance is adopted for affinity measurement. Furthermore, the authors used Particle Swarm Optimisation for updating the radiuses of the antigens. Through their simulation experiments, it is shown that the proposed algorithm achieved 99.1% true positive rate while the false positive rate is 1.9%

The authors in [59] have developed a multiple detector which consists of various types of AIS algorithms to detect attacks from the UNB ISCX Intrusion Detection Evaluation Dataset using various features such as connection duration, protocol type, service, flag type, source and destination ports and so on. Their results show the multiple-detector set artificial immune system achieved a Detection Rate of 53.34% and a False Positive Rate of 0.20%.

The authors in [60] the authors have used Genetic Algorithm with one-point crossover instead of two to find an optimal set of features before using ML algorithms for detecting attacks from the NSL-KDD dataset. Multiple popular supervised classifiers were used on the dataset such as Random Forest, Decision Trees, K-Nearest Neighbour, Naive Bayes and Bayesian Networks. The authors report Random Forest as the algorithm producing the best results. The best ML algorithm was Random Forest with 99.87% DR.

1.4.8 Chaos Theory Studies

Most of the studies making usage of chaos, also use ALRFC for the effective malware and intrusions detections. Below, we briefly present some notable studies that combine Chaos Theory with ALRFC.

In [61], the authors have created a (D)DoS detection system to detect (D)DoS attacks but also be robust against legitimate bursts of traffic. They aimed to discover patterns that can define how legitimate traffic behaves. Based on their observations they developed a baseline to use it to train their neural network model to detect (D)DoS attacks.

Their system uses self-similarity theory to construct a self-similar neural network model to monitor the traffic and Chaos Theory's local lyapunov exponents to distinguish between normal behaviour and (D)DoS attack. Normal and attack traffic are represented in a space graph. These points are the mappings of a non-linear function of the input variables, which includes sequences of normal, bursty normal and normal changed to attack traffic. It is assumed that legitimate traffic diminishes asymptotically with time. However, if traffic behaves chaotically as soon as new traffic enters the space then it is likely there is a (D)DoS attack. They report a detection rate of 88-94 % with a false positive of 0.45-0.05 %.

Based on the findings and the methods used in [61], the authors in [62] have followed a similar approach, managing to imrpove the detection rate. In detail, they have once again used chaos theory's lyapunov exponents in a combination with neural networks to detect (D)DoS attacks. They report a 98.4% detection rate, however they don't give any details regarding any false positive or false negative rates.

In [63], the authors again use the local Lyapunov exponent in combination with the ALRFC to detect a (D)DoS attack from the DARPA Dataset. They have used information regarding the packets flow to differentiate between (D)DoS traffic and normal burst traffic. With the results they gained they trained their Neural Network model to correctly detect (D)DoS attacks. The chaos theory model they used the AR time model to predict the network traffic and calculate the error rate. They used DARPA dataset to evaluate their work and claimed 93.75% detection rate.

1.5 Case Study: Application Layer (D)DoS Detection

Smart Homes consist of a great number of different devices, all deployed in a single network monitoring the environment, collecting and sharing important data and information with the owners and other smart IoT devices and external services through internal and external networks. The node responsible for this communication is the Energy Services Interface (ESI). It acts as a bi-directional interface where information can be exchanged between the Smart Home and external do- mains. Furthermore, it protects internal energy resources from security failures and ensures secure internal communication between the devices deployed in the Smart Home. ESIs importance to the Smart Home and in the outside domains makes it an excellent target for cyber attacks.

(D)DoS attacks can be conducted across all the layers of the TCP/IP model. Application layer (D)DoS attacks are much more harder to be detected efficiently and accurately than their perspective ones in lower layers as they do not violate any protocol rules or make usage of malicious behaviour. The TCP connections are established successfully and normal requests are sent to the target, in contrast to (D)DoS attacks in lower layer such as the TCP Flooding which sends a burst amount of SYN packets without acknowledging the SYN,ACK packets sent from the server. The Application Layer Flooding instead sends a burst amount of legitimate requests to the server, which the server cannot refuse but to reply. As a result, it becomes unresponsive due to great amount of incoming requests. On the contrary, Slow-Rate Application Layer (D)DoS attack exploits a servers ability to wait for a connections to be completed in a range of time, if the incoming connection is legitimately slow. As long as the client manages to send a subsequent packet in an attempt to complete the request the server is obliged to keep the connection open. Based on that, the slow-rate attack opens a great number of connections and initiates requests that never completes them. As time is passing by, more and more connections are open and that results in the target becoming once again unresponsive.

Based on these issues and concerns, it is important to develop effective security countermeasures to detect (D)DoS attacks in such critical systems. However, the majority of devices deployed in IoT networks are low in resouces such as memory, processing power and power. Therefore, it is important for the detection algorithms to be as lightweight as possible. Since, the most popular detection algorithms used belong to Artificial Intelligence and Machine learning it is even more important since they are particularly high in complexity and need a large dataset to be trained. One effective way to reduce an algorithm's complexity is to reduce the features. In that way, the algorithms are going to be trained faster, in less time and the model constructed is going to be less complex. nTherefore, we have constructed an Smart Home IoT dataset which consists of both legitimate and Application Layer Flooding and Slow-Rate (D)DoS attacks. The dataset consists of 197 attack instances and 257 normal instances.

1.5.1 Evaluation Environment

1.5.2 NS-3 Network Simulator

For the simulation of the Smart Home network, Network Simulator 3 (NS3) was used. Network Simulator 3 (NS3) is an open-source discrete-event simulator developed in C++. A summary of the networks parameters and protocols used is given in Table 1.2 . Various IoT devices are deployed in the Home Area Network that connect to the gateway, called ESI, to exchange data with the Smart City infrastructures. For our IoT Smart Home network simulation, 10 nodes were simulated. 9 of them represent IoT devices and Smart Appliances and the final node forms the ESI. All simulated nodes were static, with no mobility. In the physical layer, all the connections were wireless and the 802.11 protocol was used. The structure was set to ad hoc, using AODV routing protocol . In the network layer IPv4 was used and in the transport layer both UDP and TCP were used. In the application layer a wide variety of application layer protocols were simulated including CoAP, MQTT, XMPP and AMQP and HTTP.

1.5.3 Simulation of (D)DoS attacks

Our proposed algorithms accuracy is evaluated through experiments. Related (D)DoS detection studies make usage of popular datasets such as the KDD99, DARPA 1998 or NSL-KDD Datasets. However, we cannot use these datasets as they do not have IoT traffic nor they contain application layer (D)DoS attacks. In IoT networks the traffic is highly heterogeneous, therefore harder to detect any malicious behaviour. Hence, we had to create our own synthetic traffic using NS-3. A series of traffic containing normal and attack traffic files were generated. Both flooding and slow-rate attacks were simulated. In every scenario, seven nodes from the Smart Home were generating normal traffic and the remaining two were generating malicious traffic.

Table 1.2 Smart Home Protocols

Layer	Protocol SH/SG-Substations/SG-AMI
Physical	802.11
Data-Link	SSID
Network	IPv4
Transport	TCP
Application	HTTP, MQTT, CoAP, XMPP

In order for both normal and attack traffic to be simulated, degree of randomness was added in the network packet gen- eration from the nodes. Randomness was introduced through Poisson Distribution in the following metrics, at the time a packet was created and sent from the client to the server, at the size of the packet generated from both the client and the server and at the time the server needed to respond. Different speeds were also applied in IoT nodes to service application layer requests. Hence, we can simulate slow connections that are legitimate.

Most of the (D)DoS attacks, Flooding types in particular, can be classified as constant rate. In constant rate attacks, the attackers generate a high steady rate of traffic towards the target [5]. The impact of such an attack is fast, but it can easily be detected due to the its obvious intensity. Hence, attackers have moved towards more sophisticated ways of conducting (D)DoS attacks. One way to evade any security measures installed is to slowly but steadily flood the target in an increasing-rate attack, where the maximum impact of the attack is reached gradually over the attack period. To simulate increased-rate attacks in NS-3, we used open random connections after random time-intervals. To assess the results on ForChaos algorithms effectiveness we have calculated the most popular metrics used when assessing a detection algorithms accuracy.

These are Detection Rate(DR), Error Rate (ER), True Positives(TP), False Positives (FP), False Negatives (FN) and Precision. In intrusion detection positives instances are attacks and negative instances are normal. DR measures the algorithms proportion in correctly classifying incoming instances and ER measures the algorithms proportion errors incorrectly classifying incoming instances. DR and ERs sum equals one. TP measures the proportion of positive instances that are correctly identified as such. FP measures the proportion of negative instances to have been misclassified as positive. FN measures the proportion of positive instances that have been misclassified as negative. Lastly, precision measures the proportion of relevant instances among the retrieved instances. Hence, precision measures the rate of true positives divided by all the positives, both correctly and incorrectly classified.

1.5.4 Feature Selection

Most of the studies using forecasting techniques against (D)DoS attacks mentioned in the previous section usually make usage of a single feature for prediction and detect the possible deviations in the number of packets, the number of packets per IP, the packet flags and so on. However, these studies focus on detecting (D)DoS

Table 1.3 List of Features

Feature Name	**Description**
Requests No	Total number of requests in a time-series interval
Packets No	Total number of packets in a time-series interval
Data Rate	Average data rate in a time-series interval
Avg Packet Size	Average packet size in a time-series interval
Avg Time Betw Requests	Average time between two requests in a time-series interval
Avg Time Betw Response & Request	Average time between the response and the first requests encountered in a time-series interval
Avg Time Betw Responses	Average time between two responses in a time-series interval
Parallel Requests	Total number of parallel requests in a time-series interval

attacks on lower layers and not on the application layer. The adoption of one single feature on the application layer is likely not to produce satisfactory results since Application (D)DoS attacks do not violate any protocol rules and do not produce malformed packets. Therefore we have designed a new set of features to detect Application Layer Flooding and Slow-rate (D)DoS attacks. All of the features are heavily dependent on time and form ideal features for forecasting-based algorithms and presented in Table 1.3.

1) Requests Number: In a flooding scenario the number of requests over a period of time will be much greater compared to normal traffic request rates. In a slow-rate attack scenario the number of requests over two consecutive periods of time will have large difference. Since slow-rate opens connections to send requests after an amount of minutes there will be a pattern of low number of requests, then high number of requests, then low and so on.

2) Packets Number: In a flooding attack the number of packets on application layer is rapidly increased over a short period of time. In a slow-rate attack the number of packets is lower due to the packets being sent slowly, just before a request can be rejected. Hence, the request stays alive but very few packets are sent.

3) Data Rate: In a flooding attack the data rate on application layer is rapidly increased over a short period of time. In a slow-rate attack the data rate is lower due to the slow data rate of the malicious requests.

4) Average Packet Size: In a flooding attack the average packet size is decreased due to the requests being simply a get request with no payload. In a slow-rate attack the average size packet is decreased since a great series of connections send small incomplete packets over the attack period of time.

5) Average Time between Requests: In a flooding attack the average time between each request is vastly decreased due to an outburst of requests being sent over a short period of time simultaneously. In a slow-rate attack the average time between each request is decreased compared to the same value in a normal scenario. Instead, there are spikes of the number of requests being sent. When the attack is at the stage of opening new requests, their rate is increased. When the attack is at the stage of maintaining the connections open, normal subsequent packets are sent, therefore, the rate of the requests is less than the previous stage.

6) Average Time between Response and Request: In a flooding attack the average time between a response and the next request is vastly decreased. In a slow-rate attack the average time between a response and the next request the slow-rate attack causes malicious requests.

7) Average Time between Responses: In a flooding attack, the average time between two consecutive responses is vastly decreased since a burst of requests is being sent to the server, therefore a great number of responses have to be generated. In a slow-rate attack, the same feature is increased since the slow-rate attack's requests are initiated but never completed.

8) Parallel Requests: In a flooding attack the average concurrent requests is decreased since the attackers send multiple requests that are very quickly finished. In a slow-rate attack, many requests are initiated and stay open over a period of time and the average number of parallel open requests is large.

1.5.5 Intrusion Detection Evaluation Metrics

In the Intrusion Detection Literature, studies use an extended series of evaluation metrics and statistical tests to asses their detection engines performance. Measuring the system's detection rate is not enough sometimes, therefore to provide a more hollistic evaluation, different metrics are adopted. In this section, we briefly describe the most popular evaluation metrics, the actual formulas and their application in Intrusion Detection. In Intrusion Detection research, the presence of a cyber attack is regarded as a positive instance and the absence of it as a negative instance.

True Positive Rate

True Positive Rater (TPR) or otherwise called Recall or Sensitivity, is when the detection algorithm is able to correclty identify an instance as an attack.

True Negative Rate

True Negative Rater (TNR) or otherwise called Specificity, is when the detection algorithm is able to correclty identify an instance as normal.

False Positive Rate

False Positive Rate (FPR), or otherwise called Fall-Out, is when the detection alglorithm has incorrectly classified an instance as an attack, with it being normal.

False Negative Rate

False Negeative Rate (FNR) is when the detection alglorithm has incorrectly classified an instance as normal, with it being an attack.

Precision

Table 1.4 Machine Learning Algorithms Results against Application Layer (D)DoS Attacks Smart Home IoT dataset with 8 Features

Alg (Dataset)	**DR (%)**	**TP (%)**	**FP (%)**	**TN (%)**	**FN (%)**	**Prec (%)**
NB(1)	89.54	83.8	5.9	94.1	16.2	91.9
BN(1)	91.50	80.9	0	100	8.50	100
SVM(1)	88.24	73.5	0	100	26.5	100
KNN(1)	98.69	97.1	0	100	1.31	100
DT(1)	95.42	97.1	5.9	94.1	2.9	93
RF(1)	96.73	97.1	3.5	96.5	2.9	95.7
MLP(1)	98.69	98.5	1.2	98.8	1.5	98.5
DeepMLP(1)	95.42	89.7	0	100	10.3	100

Precision, or otherwise called positive predictive value, denotes the rate at which the detection algorithm has correctly classified True Positive instances out of all the positive instances it has classified. In general, Precision denotes the fraction of instances that are relevant from the total number of all retrieved instances.

1.5.6 Experiments Results Analysis

As it has been highlighted in [60, 47, 46, 49, 39, 40, 41], bio/nature-inspired algorithms have been used for feature reduction in an attempt to reduce the overall compexity of detection algorithms without affecting their accuracy. For finding an optimal set of features, we have used the available nature-inspired algorithms provided from Weka toolkit. Specifically, we used evolutionary search, ant and bee search, genetic search and particle swarm optimisation search algorithms. All algorithms have identified Parallel Requests, Average Data Rate, Average Packet Size and Packet Number as the necessary features. The Machine Learning algorithms used were Random Forests (RF), Decision Trees (DT), K-Nearest Neighbour, SVM, Bayesian Networks, Naive Bayesian, Multi-Layer Perceptron Artifcial Neural Network, Deep Multi-Layer Perceptron (MLP) Artifcial Neural Network, Two series of experiments have been conducted, one with all eight features and one with four features.

1.5.7 Results Analysis and Discussion

From our experiments, it was observed that the overall detection rate has been vastly decreased across all algorithms when the features were reduced. The specific rates are enlisted in Table 1.4 (with 8 features) and in Table 1.5 (with 4 features).

Detection rate: MLP and KNN have performed best with 8 features, both with 98.69%, followed by RF with 96.74%,DT and Deep-MLP with 95.42%, and then BN with 91.50%, NB with 89.54% and SVM with 88.24%.

In the series of experiments using 4 features KNN has still been the most accurate in classifying unknown traffic but MLP's detection rate has been dropped. In detail, KNN comes first in terms of detection rate, followed by SVM, MLP, BN,

Table 1.5 Machine Learning Algorithms Results against Application Layer (D)DoS Attacks Smart Home IoT dataset with 4 Features

Alg (Dataset)	**DR (%)**	**TP (%)**	**FP (%)**	**TN (%)**	**FN (%)**	**Prec (%)**
NB(1)	86.61	82	7.8	92.2	18	92.6
BN(1)	89.29	85.2	5.9	94.1	14.8	94.5
SVM(1)	89.29	82	2	98	18	98
KNN(1)	93.75	93.4	5.9	94.1	6.6	95
DT(1)	88.39	88.5	11.8	88.2	11.5	90
RF(1)	88.39	90.2	13.7	86.3	9.8	88.7
MLP(1)	89.29	86.9	7.8	92.2	13.1	93
DeepMLP(1)	87.5	83.6	7.8	92.2	16.4	92.7

DT, RF, Deep-MLP and NB. Based on the results, we can observe that KNN's distance metrics is accurate enough regardless of the features involved for effectively differentiating between normal and attack traffic.

The reduction of features clearly negatively affects MLP, Deep-MLP, RF and DT's accuracy. The feature reduction makes the ALRFC models' characteristics such as the connections between the layers, weights and number of hidden layer in the Deep-MLP less complex but also less accurate. The model generated is too simplistic to identify the kind of incoming traffic. Also, the reduction of features makes DT and RF models to create less compelx trees that has the same negative effect. On the other hand the probabilistic models used, BN and NB, are not particularly negatively affected due to joint probabilities used. However, with the feature reduction the SVM algorithm's detection rate is vastly increased. This could be because the hyperplane constructed with less features is actually more accurate than with 8 features.

True Positive Rate: In the 8 features series of experiments, MLP achieves the best TP rate with 98.5%. KNN RF and DT have the highest TP rates, all with 97.1%. RF performed better than DT as expected due to RFs being an improved DT version. The algorithms following are Deep-MLP with 89.7%, Naive Bayes 83.8%, Bayesian Networks with 80.9% and SVM with 73.5%. The Deep-MLP model although having a high TP, rate it can be highlighted that the construction of additional hidden layers does not necessarily mean they are going to be more accurate in detecting attacks so an MLP with a single hidden layer is a better solution. Also, it is indicated that Probabilistic models and SVM have not been particularly effective in constructing models to detect the attacks accurately enough.

In the 4 features series of experiments, the TP rate is generally decreased across all algorithms. Only KNN and RF manage to get a TP rate above 90 with 93.4% and 90.2%. Then DT follows with 88.5%, MLP with 86.9%, BN with 85.2%, Deep-MLP with 83.6% and lastly NB and SVM with 82%.

It is worth highlighting that the feture reduction vastly decreases the algorithms' general accuracy and ability to detect the attacks. In sophisticated attacks such as the Application Layer (D)DoS attacks, features play an important role in the construction of the algorithmic models. Based on both of these series of experiments, it is evident that probabilistic approaches fail to identify the attack instances as BN and NB fail to get a TP rate above at least 90%. This occurs because probabilistic models need a large dataset to construct accurate and robust probabilities. Furthermore due to the small dataset as well as the feature reduction, MLP and Deep-MLP failt to create robust algorithmic models. On the other hand, KNN proves to be robust in detecting unknown attacks along with RF and relatively DT.

False Positive Rate: In the series of 8 features BN, SVM, Deep-MLP and KNN do not generate any FP instances. MLP generates a 1.2% FP rate, RF generates a 3.5% FP rate, and DT and NB generate 5.9% FP rate . In the series of 4 features SVM generates 2% FP rate, BN and KNN generate 5.9% FP rate , MLP, Deep-MLP and NB generate 7.8% FP rate , DT generates 11.8% and RF 13.7% FP rates. In general, it is observed that with the reduction of features the FP rate is increased. This indicates that the 4 features removed are quite important in understanding when normal behaviour can have anomalies but still be normal.

1.6 Discussion

Without a doubt, the work done in the field of intrusion detection using bio-inspired techniques is remarkable. There is a great diversity of different techniques used to both reduce the features and/or detect the various types of malicious behaviour. Bio/Nature-inspired algorithms have been proven particularly effective in finding an optimal set of features that can achieve high rate of detection as well as a low rate of false positives. This is particularly beneficial as it reduces the algorithmic complexity of the detection model constructed as well as the time complexity. Furthermore, through the studies it is highlighted that bio/nature-inspired algorithms are particularly effective in detecting different types of attacks that may or may not belong in the same attack family, such as the KDD99 Dataset.

As dsicussed above, studies achieve results of detection rate above 77% and false positive rates not below 2.7% as illustrated in Table 1.6.

Artificial Neural Networks are proved the most effective in terms of the detection technique , achieving the highest detection rate with 99.72%. This is reflected in the research literature as ANN is proved to be the most usable and popular bio-inspired algorithm in intrusion detection.

However, ALRFC have an increased computational complexity [39]. To decrease and optimize the complexity threshold techniques, such as Genetic Algorithms and Chaos Theory, could be applied . Hence, ANN will need fewer resources.

Game Theory is mainly used for preventing intrusions in systems. However, its ability to define how network nodes in the system should communicate makes it a suitable approach to detect malicious behaviour. For that reason the authors in [14],[15] have applied a combination of computational puzzle with game theory to prevent an attack rather than just detecting it. Game Theory can also be used for trust management between the nodes participating in detection of threats in the system.

Genetic Algorithms have a better detection rate as shown in [36,37,38] than Evolution Strategies [35] although both technologies follow a similar apporach. GA algorithms are behaving effectively, from resources allocation and execution time [18], for feature reduction. In next generation, GA algorithms can be more effective as the dataset can be diverse and provide better detection with fewer errors. This is due to packet dropping, which is normal in communication networks, and it cannot be considered an attack unless it is examined more carefully. GA algorithms are behaving effectively for feature reduction before intrusion detection. Regarding detection, GA is able to achieve a detection rate between 90-97.57% for KDD99 attacks. This indicates that the Evolutionary approach GA follows is effective against detecting different types of attacks as well as different variations of the same attack. Also, GA is able to detect other types of (D)DoS attacks such as Grayhole and packet

Table 1.6 Bio/Nature-Inspired Algorithms Studies Results

Algorithm/ Ref.	**Attack**	**Result**
GT [32]	(D)DoS Attacks	Prevention
GT [33]	(D)DoS Attacks	Prevention
EA [34]	SIP Flooding Attacks	82.17% and 77.00%
ES [35]	Darpa (D)DoS attacks	No Results
GA [36]	Blackhl. & Packt. Dr.	95%
GA [37]	KDD99 Attacks	90%
GA [38]	KDD99 Attacks	97.57%
GA & FL [39]	KDD99 Attacks	95%
GA & FL [40]	KDD99 Attacks	95% and 1% FPR
GA & FL [41]	KDD99 Attacks	99.75%
FL [42]	Jamming DoS	99.75%
ACO [43]	UDP DoS	80%
ACO [44]	Low-rate DoS	89%
ACO [45]	KDD99 Attacks	96.94%
ABC & ANN [46]	KDD99 Attacks	less 0.025 squar er.
GA & ANN [49]	KDD99 Attacks	99.997% DR and 0.002% FPR
ANN [50]	KDD99 Attacks	99.723% DR and 0.277% FPR
ANN [51]	NSL-KDD Attacks	96.6% DR and 3.4% FPR
ANN [52]	KDD99 Attacks	False Accept. less 8% & False Rej. less 5%.
ANN [53]	(D)DoS	85% TPR
AIS [54]	DARPA (D)DoS	max TN 8.5% and max FN rate 15%
AIS [55]	KDD99	FP 0.7% & FN 21.02% & 1.3% and 26.32%
AIS [56]	SYN Attacks	100% TPT & 0.17 FPR
CT [61]	(D)DoS Attacks	88-94 % & .45-0.05 % FPR
CT [62]	(D)DoS Attacks	98.4%
CT [63]	DARPA Attacks	93.75%
PSO with FLN [47]	KDD-99	Norm. 99.7%, (D)DoS 98.11%, U2R 80.4%,R2L 54.55%
AIS [57]	NSL-KDD	] 86.8%
AIS [58]	KDD-99	TPR 99.1% and FPR 1.9%
AIS [59]	KDD-99	DR 53.34% and FPR 0.20%.
GA with RF [60]	NSL-KDD	99.87% DR.

dropping attacks as shown in [18] with a high detection rate (95%). GA can also be used in combination with other bio/nature-inspired algorithms such as [39, 40, 41, 49]. In suc case, GA is used for feature reduction in an attempt to reduce complexity of complex architectures (ANN and FL) without affecting the detection rate. This objective is achieved as the detection rate is between 95-99.997%.

Artificial Immune Systems are particularly effective against intrusion detection in distributed systems since there is no need for central coordinator. In [54] authors detect (D)DoS attacks based on the AIS algorithm which is able to spread itself across the network for monitoring. Therefore, (D)DoS attacks are detected faster since the proposed Negative Search Structure algorithm is able to mix antibodies and detect anomalies. In addition, from the studies of [54,55,56], it is shown that their proposals produce low false positive rates since authors construct systems that can immediately detect abnormal instances. Hence, AIS algorithm is able to detect a wide range of (D)DoS attacks as well as other types of attacks, when KDD99 is used. However, AIS has a relatively high false negative rate, between 15-26.32% whereas the false positive rate remains low, between 0.17-1.3%.

SI achieve a respectable detection rate against a range of cyber attacks as they take advantage of the algorithm's cooperative nature to detect various types of attacks that can be considered strealthy, such as low-rate (D)DoS attacks[44] and UDP DoS attacks [43], achieving a good detection rate of 89% and 80% perspectively. In the KDD99 attacks, SI algorithm is performing well while achieving a 96.94% detection rate.

Fuzzy Logic also achieves high rates of detection as shown in [39, 40, 41, 42]. Fuzzy Logic is also effective in comparing classifications from many sources and deciding if an instance is either malicious or normal. FL promotes collaborative detection by effectively combining evidence from multiple IDS agents. Therefore, it is able to detect the multiple and different cyber attacks contained in the KDD99 dataset with a detection rate between 95-98.3%. FL in [54] has also been used against Jamming DoS attacks with a very high detection rate of 99.75%.

Finally, Chaos Theory is proved to also be a promising approach in detecting abnormalities in real time traffic as shown in [61,62,63] when combined with AL-RFC achieving a detection rate between 88-98.4% against various (D)DoS attacks in different protocols.

Although the studies on bio-inspired techniques and their effective application in intrusion detection seem more than promising, they certainly have some limitations. Cyber-attacks have become highly sophisticated and an attacker can stealthy bypass any security control.

It can be observed that current studies discuss how robust against cyber attacks and (D)DoS attacks in particular. However, there is no indication on how to detect compromised nodes/botnets acting maliciously. As a counter-example, an attacker can constantly change IP addresses to bypass (D)DoS attack detection. In a similar example, botnets can flood the network then sleep for a while and wake again to carry on so as to avoid detection. There are no studies that target such scenarios. This issue should have been at least considered since many (D)DoS attack scenarios are initiated from one machine being infected and gradually infecting the rest of the network so they can all become a part of a large botnet.

Furthermore, the U2R and R2L scenarios included in the KDD-99 Dataset that have been used in multiple studies. Although these malicious types of behaciour are useful for detecting more complex cyber attacks scenarios none of the related studies make usage of more complex cyber attack scenarios such as APTs. Hence, more sophisticated and recent datasets shall be used to evaluate bio-inspired algorithms.

Unfortunately, only a few studies exist that make usage of bio-inspired algorithms in the cyber physical systems [42,51,52,53,55]. Other techniques could also be used to detect malicious behaviour in other cyber-physical systems such as the Smart Grid, IoT networks and the Smart City. Firstly, Fuzzy Logic could be used and be particularly effective. As seen from the studies of [39, 40, 41, 42], fuzzy logic promotes collaboration, an essential concept for the detection of advanced persistent attacks in distributed systems such Smart City networks.

Such systems are heterogeneous and complex, therefore, multiple information must be collected from different intrusion detection sensors to identify if there is malicious behavior present. Another method that could be proved particularly effective for these infrastructures is the Swarm Intelligence algorithm since it also promotes cooperation and it can be particularly effective against unknown attacks, or anomaly detection. Chaos Theorys strong advantage is that it can capture legitimate behaviour and define how much alteration can still be considered legitimate. Also, Chaos Theory could be used to detect other types of attacks that are relatively new, such as the Application Layer (D)DoS attacks. Lastly, Game Theory would also be an effective solution for preventing (D)DoS attacks.

In Smart City IoT networks and critical infrastructures there is a vast amount of intelligent electronic devices interconnected together spread into large geographical distances. Due to their close mutual operation they have higher possibilities of being compromised and participate in botnet attacks.

Another observed limitation, is that researchers do not consider compromised IDS agents before actually launching the DoS or (D)DoS attacks. If one or more IDS agents are compromised, the attacker violates the trust between the IDS agents. Hence the compromised agent can fool the network and act as legitimate. The only studies that address this issue are [32] and [33]. Such scenarios are critical to the system and can cause major consequences.

Additionally, bio-inspired techniques are proved to be particularly effective against (D)DoS attacks as it can be seen from the studies. We still believe they can be just as effective against other types of (D)DoS attacks, such as the slow rate attack in the application layer [42]. Low rate and slow rate attacks follow similar approach but take place in different OSI Layer. Hence, detection can be efficiently achieved using bio-inspired techniques. Additional studies and extended experiments must be conducted for such an assumption to be validated. In addition, other (D)DoS attacks can be explored, such as the DNS Amplification attack;

There have been studies to defend systems against application layer (D)DoS attacks but none of them have made usage of bio-inspired techniques or has examined how to defend against such attacks in the IoT networks and critical infrastructures

[53, 54, 55, 56]. However, since Smart Grid communicates on application level as well, such attacks are possible to be conducted on such environment. Bio-inspired techniques could be particularly effective in the Smart Grid due to its collaborative nature.

Another direction to consider effective detection of cyber attacks in the Smart Grid is the possibility of an attack being initiated from another infrastructure still targeting the Smart Grid. One example of such a circumstance is Smart Homes and the Internet of Things. Smart Homes are directly connected to the Smart Grid. It is much easier for an adversary to compromise a number of Smart Home devices and attack the Smart Grid than trying to harm the Smart Grid directly.

An important aspect to consider is that the majority of devices deployed in the Smart Grid are low in resources. On the other hand most of the bio-inspired algorithms are resources demanding. Therefore, possible changes to the algorithms could be considered to make them lightweight and possibly adapt to such devices.

In the future several bio-inspired techniques may combined with other machine learning techniques, such as in [41] with SVM. Mitigating approaches rather than just detection be considered in the near future. Additionally, new solutions could expand to hybrid attacks and increase security confidence in networked systems.

Although the techniques used in intrusion detection are wide in range, there is still room for introducing new bio-inspired techniques as they could detect anomalies in a given set using nature-inspired models. An example is the self-healing technique used in the field of organic computing. In nature, self-healing is the automated process of recovery of a natural being. In research, most likely bioinformatics, the aim is to construct the self-X properties set depending on the goal. A self-detection approach could be designed as inspired by [37, 38].

Another potential approach is the Firefly Algorithm, inspired by the fireflies behavior. A fireflys main behavior is flashing so it can attract other fireflies. The concept is that the brighter the colour is, the greater the attractiveness is. Distance also plays a role as it increases brightness. Firefly Algorithm could be potentially used for feature selection and reduction as well as used as an optimization technique for ANN/SVM/Fuzzy Logic techniques.

Besides detection, possible mitigation/prevention methods should be explored. A good example for this process is the Cuckoo search optimization technique as introduced in Section II. It is motivated by the cuckoo species that let their eggs on hosts of other birds. If birds understand that their eggs are not their own they throw them away or build a nest somewhere else. Eggs represent solutions and a cuckoo egg consists of a new solution that can potentially replace the old one. The algorithm can grow to a complex solution consisting a series of solutions. After detection, it can be used to find an optimized solution. Also, Cuckoo technique could be used for feature reduction and possible weight training of ALRFC.

1.7 Conclusion

As the technology rapidly progresses, it becomes a vital part of our lives with the Smart City era approaching. Critical infrastructures and IoT networks become more complex in the Smart City context but still need to be continuously available. Hence, the protection of infrastructures, such as the Smart Grid, against (D)DoS attacks is important. Since an attack can become highly sophisticated the usage of natural systems to protect networks from intrusions could transform intrusion detection systems into fast, accurate and robust systems able to effectively detect plethora of (D)DoS attacks. Bio-inspired techniques are proved to be an efficient and practical solution to various attacks.

In this chapter, we discussed the importance of bio-inspired techniques in the field of intrusion detection. A brief explanation was given on the most popular cyber attacks, as well as the most popular concepts of bio-inspired algorithm detection. Furthermore, the most recent and high impact studies on this topic have been presented in a wide range of networks and a detailed discussion was made regarding these studies. Concluding, future directions were highlighted for new research exploration in the bio-inspired era.

References

[1] Elisan, Christopher C., & Mikko Hypponen. Malware, rootkits & botnets: A beginner's guide. McGraw-Hill, (2013).

[2] S. Olariu, A. Y. Zomaya, Handbook of Bioinspired Algorithms and Applications, Chapman and Hall/CRC , 2005.

[3] J. Heaton, Artificial Intelligence for Humans, Volume 2: Nature-Inspired Algorithms. CreateSpace Independent Publishing Platform, 2014, pp.1-38

[4] McCulloch, Warren; Walter Pitts . "A Logical Calculus of Ideas Immanent in Nervous Activity". Bulletin of Mathematical Biophysics, volume 5 (4) pp. 115133, 1943.

[5] Eiben, Agoston E., & James E. Smith. Introduction to evolutionary computing. Vol. 53. Berlin: springer, 2003.

[6] J.H. Holland, Genetic algorithms and the optimal allocation of trials, SIAM J. Comput. 2 (2) pp. 88105, 1973.

[7] Koza, John R. Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA: The MIT Press, 1992.

[8] Beyer, H.G. and Schwefel, H.P. " Evolution strategies ". Natural Computing volume 1, pp. 352, 2002.

[9] Binitha S. , S S. Sathya, "A Survey of Bio inspired Optimization Algorithms", International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-2, May 2012.

[10] Fogel, Lawrence J., Alvin J. Owens, & Michael J. Walsh. "Artificial intelligence through simulated evolution." (1966).

[11] Beni, G., Wang, J. "Swarm Intelligence in Cellular Robotic Systems", Proceed. NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy, June 2630, 1989.

[12] Kennedy, J.; Eberhart, R. . "Particle Swarm Optimization". Proceedings of IEEE International Conference on Neural Networks. IV. pp. 19421948, 1995.

[13] Dorigo, M., Maniezzo, V., & Colorni, A. ”Ant System: Optimization by a colony of cooperating agents” . IEEE Transactions on Systems, Man, and Cybernetics Part B, 26, pp. 2941, 1995.

[14] Karaboga, Dervis. An idea based on honey bee swarm for numerical optimization. Vol. 200. Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005.

[15] X. Li, Z. Shao, J. Qian, ”Anoptimizing method base on autonomous animates: fish-swarm algorithm” , Systems Engineering Theory and Practice volume 22 (2002) pp. 32 38, 1995.

[16] X.S, Yang. ”Firefly algorithms for multimodal optimization.” International symposium on stochastic algorithms. Springer, Berlin, Heidelberg, 2009.

[17] - Beyer, H.G. and Schwefel, H.P. ”Evolution strategies. Natural Computing” volume 1, pp.3 52, 2002.

[18] Zadeh, L.A. ”Fuzzy sets”. Information and Control, volume 8 (3), pp. 338353, 1965.

[19] Lorenz E. Predictability: Does the flap of a butterflys wings in Brazil set off a tornado in Texas. American Association for the Advancement of Science, Washington, DC. 1972.

[20] Neumann, J. V., ”Zur Theorie der Gesellschaftsspiele”, Mathematische Annalen, 100 (1): 295320, doi:10.1007/BF01448847 English translation: Tucker, A. W.; Luce, R. D., eds. (1959), ”On the Theory of Games of Strategy”, Contributions to the Theory of Games, 4, pp. 1342, 1928.

[21] McDowell, M. ”Understanding Denial-of-Service Attacks US-CERT.” United States Computer Emergency Readiness Team (2013).

[22] Elisan, Christopher C., & Mikko Hypponen. Malware, rootkits & botnets: A beginner’s guide. McGraw-Hill, (2013). pp 9-82.

[23] DARPA intrusion detection evaluation.

Available on: $http://www.ll.mit.edu/IST/ideval/data/data_index.html$

[24] KDD Cup 1999.

Available on: $http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html$

[25] NSL-KDD.

Available on: $http://www.unb.ca/cic/datasets/nsl.html$

[26] Shiravi, A., Shiravi, H., Tavallaee, M. and Ghorbani, A.A.,” Toward developing a systematic approach to generate benchmark datasets for intrusion detection”, computers & security, 31(3), 2012 pp.357-374.

[27] Sharafaldin, I., Lashkari, A.H. and Ghorbani, A.A., ”Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization”. In ICISSP, 2018, pp. 108-116.

[28] Beigi, E.B., Jazi, H.H., Stakhanova, N. and Ghorbani, A.A., ”Towards effective feature selection in machine learning-based botnet detection approaches.” In Communications and Network Security (CNS), 2014 IEEE Conference on, 2014, pp. 247-255.

[29]Jazi, H.H., Gonzalez, H., Stakhanova, N. and Ghorbani, A.A., ”Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling.” Computer Networks, 121, 2017, pp.25-36.

[30] Kolias, C., Kambourakis, G., Stavrou, A. and Gritzalis, S., "Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset. IEEE Communications Surveys & Tutorials, 18(1), 2018, pp.184-208.

[31] Moustafa, N. and 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. Information Security Journal: A Global Perspective, vol. 25 no. 1-3, 2016 pp.18-31.

[32] Q. Zhu, C. Fung, R. Boutaba, and T. Baar, Guidax: A game-theoretic incentive-based mechanism for intrusion detection networks, IEEE J. Sel. Areas Commun., Special Issue on Economics of Communication Networks & Systems (SI-NetEcon), 2012 pp.2220-2230.

[33] A.Michalas, N.Komninos,N.R.Prasad, Multiplayer Game for (D)DoS Attacks Resilience in Ad Hoc Networks. in 2nd International Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, pp.1-5, 2011.

[34] M. A. Akbar and M. Farooq. Application of evolutionary algorithms in detection of SIP based flooding attacks. In Proc of the 11th Annual conference on Genetic and evolutionary computation, GECCO 09, pages 14191426, 2009.

[35] Herve Kabamba Mbikayi, An Evolution Strategy Approach toward Rule-set Generation for Network Intrusion Detection Systems (IDS), International Journal of Soft Computing and Engineering (IJSCE), vol. 2 Issue-5, pp. 201205, 2012.

[36] M.Lali, V. Palanisamy, Intrusion detection for MANET to detect unknown attacks using genetic algorithm. in IEEE International Conference on Computational Intelligence and Computing Research, pp. 1-5, 2014.

[37] M.Padmadas et.al., Layered Approach for Intrusion Detection Systems based Genetic Algorithm., in International Conference on Computational Intelligence and Computing Research (ICCIC), pp.1-4, 2013.

[38]Khan, F.H., Shams, R., Aamir, M., Waseem, M. and Memon, M., "Intrusion detection in wireless networks using Genetic Algorithm." In Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on, 2015, pp. 1830-1835.

[39] Y. Danane, T. Parvat, Intrusion Detection System using Fuzzy Genetic Algorithm. in International Conference on Pervasive Computing, pp. 1-5, 2015.

[40] P. Jongsuebsuk, N. Wattanapongsakorn, and C. Charnsripinyo, Network intrusion detection with fuzzy genetic algorithm for unknown attacks, inInformation Networking (ICOIN), pp. 15, Jan 2013.

[41] A. Kannan, G.Q. Maguire,Selection Algorithm for Effective Networks in IEEE 12th International Workshops,pp. 416 423, 2012.

[42] C.Balarengadurai & S Saraswathi, "A Fuzzy based Detection Technique for Jamming Attacks in IEEE 802.15.4 Low Rate Wireless Personal Area Network" in proceedings of Advances in Intelligent Systems and Computing-Springer Verlag-LNEE, pp: 422-433,2012.

[43] Gupta, A., Pandey, O.J., Shukla, M., Dadhich, A., Ingle, A. and Ambhore, V. "Intelligent Perpetual Echo Attack Detection on User Datagram Protocol Port 7 Using Ant Colony Optimization." In Electronic Systems, Signal Processing

and Computing Technologies (ICESC), 2014 International Conference on, 2014, pp. 419-424.

[44] Chen, H. H., & Huang, S. K.. ”L(D)DoS Attack Detection by Using Ant Colony Optimization Algorithms.” J. Inf. Sci. Eng., vol. 32, no 4, 2016, 995-1020.

[45] Cai, C. and Yuan, L., . ”Intrusion detection system based on ant colony system.” Journal of Networks, vol. 8, no. 4, 2013, p.888.

[46] Barani F., Barani A., ”Dynamic Intrusion Detection in AODV-based MANETs Using Memetic Artificial Bee Colony algorithm”, IEEE Conferences: 2014 22nd Iranian Conference on Electrical Engineering (ICEE), 2014, pp. 1040-1046.

[47] Ali, M.H., Al Mohammed, B.A.D., Ismail, A. and Zolkipli, M.F., ”A new intrusion detection system based on Fast Learning Network and Particle swarm optimization.” IEEE Access, 6, 2018, pp.20255-20261.

[48] Q.Qian, J.Cai, R.Zhang,Intrusion Detection based on Neural Networks and Artifical Bee Colony in IEEE/ACIS 13th International Conference on Computer and Information Science, pp.257-262, 2014.

[49] M. Barati, A. Abdullah, N I Udzir, R. Mahmod & N. Mustapha, Distributed Denial of Service Detection using hybrid machine learning techniques, International Symposium on Biometrics and Security Technologies, IEEE, 2014, pp. 268-273.

[50]L.Jin, Y.Liu, L.Gu, (D)DoS attack detection based on neural network. in International Symposium on Aware Computing, pp. 196-199, 2014.

[51] Javidi, M.M, Nattaj, M.H. A new and quick method to detect DoS attacks by Neural Networks. Journal of Mathematics and Computer Science, Volume 6, pp.85-96, 2013.

[52] H. Atashazar and H. Modaghegh, Hybrid Packet Filtering for overcoming (D)DoS Attacks against AMI components, Smart Grid Electrical Grids Technology, pp.1-5, 2013.

[53] M. Shojaei , N. Movahhedinia and B. T. Ladani, (D)DoS attack Detetion in IEEE 802.16 based networks , Wireless Networks, vol.20 (8) pp. 2543-2559, 2014.

[54] Wang, D., He, L., Xue, Y. and Dong, Y., ”Exploiting Artificial Immune systems to detect unknown DoS attacks in real-time.” In Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on Vol. 2, 2012, pp. 646-650.

[55] Y. Zhang, L. Wang, W. Sun, R. C. Green, and M. Alam, Distributed intrusion detection system in a multi-layer network architecture of smart grids, IEEE Trans. Smart Grid, vol. 2, no. 99, pp. 796808, Jul. 2011.

[56] Al-Dabagh, N.B.I; Ali, IA, ”Design and implementation of artificial immune system for detecting flooding attacks,” High Performance Computing and Simulation (HPCS), 2011 International Conference on ,pp.381,390, 4-8 July 2011.

[57] Hooks, D., Yuan, X., Roy, K., Esterline, A. and Hernandez, J., ”Applying Artificial Immune System for Intrusion Detection.” In 2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService), 2018, pp. 287-292.

[58]Tabatabaefar, M., Miriestahbanati, M. and Grgoire, J.C., ”Network intrusion detection through artificial immune system”. In Systems Conference (SysCon), 2017 Annual IEEE International, IEEE, 2017, pp. 1-6..

[59] Brown J., Anwar M., Dozier G., ”Intrusion Detection using a Multiple-Detector Set Artificial Immune System”, 2016 IEEE 17th International Conference on Information Reuse and Integration, 2016, pp. 283-286.

[60] Ferriyan, A., Thamrin, A.H., Takeda, K. and Murai, J. ”Feature selection using genetic algorithm to improve classification in network intrusion detection system.” In Knowledge Creation and Intelligent Computing (IES-KCIC), 2017 International Electronics Symposium on, 2017, pp. 46-49.

[61] A. Chonka, J. Singh, and W. Zhou, Chaos theory based detection against network mimicking (D)DoS attacks, IEEE Commun. Letters., vol. 13, no. 9, pp. 717719, Sep. 2009.

[62] X. Wu and Y. Chen, Validation of Chaos hypothesis in NADA and improved (D)DoS detection algorithm, IEEE Commun. Lett., vol. 17, no. 12, pp. 23962399, Dec. 2013.

[63] Y. Chen, X. Ma, and X. Wu, (D)DoS detection algorithm based on preprocessing network traffic predicted method and Chaos theory, IEEE Commun. Lett., vol. 17, no. 5, pp. 10521054, May 2013.