Invited Paper

SECURE RESILIENT EDGE CLOUD DESIGNED NETWORK[1]

Tarek Saadawi[2] Akira Kawaguchi Myung Lee Abbe Mowshowitz

Summary

Systems for Internet of Things (IoT) have generated new requirements in all aspects of their development and deployment, including expanded Quality of Service (QoS) needs, enhanced resiliency of computing and connectivity, and the scalability to support massive numbers of end devices in a variety of applications. The research reported here concerns the development of a reliable and secure IoT/cyber physical system (CPS), providing network support for smart and connected communities, to be realized by means of distributed, secure, resilient Edge Cloud (EC) computing. This distributed EC system will be a network of geographically distributed EC nodes, brokering between end-devices and Backend Cloud (BC) servers.

This paper focuses on three main aspects of the CPS: a) resource management in mobile cloud computing; b) information management in dynamic distributed databases; and c) biological-inspired intrusion detection system.

Keywords: *Secure, edge Cloud Network, mobile cloud computing*

1. Introduction

IoT has gained popularity in recent years and has attracted a lot of attention. Billions of smart devices have connected to IoT due to advancement of computer and networking technologies. With the advancement of cloud computing, IoT can enhance and extend its service provision capabilities. IoT can extend its scope and service provisioning capabilities with the integration of the cloud computing paradigm. On the other hand, cloud computing can enhance its services by utilizing the data collected from IoT nodes.

IoT systems have generated new requirements in all aspects of their development and deployment, including expanded Quality of Service (QoS) needs, enhanced resiliency of computing and connectivity, and the scalability to support massive numbers of end devices in a variety of applications. This paper is concerned with the development of a reliable and secure IoT/cyber physical system (CPS), providing network support for smart and connected communities, to be realized by means of distributed, secure, resilient Edge Cloud (EC) computing. This distributed EC system will be a network of geographically distributed EC nodes, brokering between end-devices and Backend Cloud (BC) servers.

Performing Complex computations on-site is possible with todays computing capacity, thus making edge computing feasible. Edge computing allows for extending cloud computing capabilities by placing services close to the network edge, thus supporting a variety of applications and services. Demand for these capabilities is being stimulated by mobile users who need more diverse types of services compared with the requirements of PC users.

The absence of centralized management dictates policies designed to maximize the use of edge nodes which typically have relatively modest storage and processing power. A possible design for realizing the benefits of non-centralized management is a system that implements hypercube routing. Such a system could support query optimization in a software defined network context, creating an engineered hypercube virtual network serving as an efficient and practical, distributed database tool.

With the rise of cyberattacks like malware, denial of service attacks, ransomware attacks or insider threats, securing computer systems has become more difficult. To improve the security of networks and computing nodes, network/computer administrators rely on protective mechanisms such as firewalls, access control, and encryption system. Another protective mechanism that can be added as a wall of protection is an intrusion detection system (IDS). Intrusion detection is defined as the process of intelligently monitoring the events occurring in a computer system

[1] This work is partially supported by USA Grant number NSF Award 1818884

[2] All four authors are with City University of New York, City College, New York, NY 10031

or network, analyzing them for signs of violations of the security policy.

The contributions of this paper are;

a) Resource Management of mobile cloud computing (MCC) in terms of efficient partitioning and offloading processes of mobile applications, an adaptive security-aware resource allocation approach that can meet the various resource requirements, and a joint multi-resource allocation in the MCC system with cloudlet.
b) Implementation of hypercube routing for query optimization in a software defined network context designed to create an engineered hypercube virtual network as an efficient and practical, distributed database tool.
c) Development of a biological-inspired intrusion detection system (IDS) for detecting attacks targetting IoT devices, the edge cloud or the cloud infrastructure.

a) Resource Management in Mobile Cloud Computing

Mobile Cloud Computing (MCC) is a system that introduces powerful Cloud Computing in a mobile computing environment, where mobile devices connect to the Internet through wireless networks and then communicate with the remote cloud. Compared to mobile devices, the cloud server of MCC can provide huge storage, high computation power, as well as reliable security [1]. By offloading subcomponents of a mobile application to the cloud server for execution, the performance of mobile applications can be greatly improved and the energy consumption of mobile devices can be significantly reduced [2]. Consequently, MCC can extend its scope to a great variety of mobile applications such as virus scanning that are extremely resource-intensive to execute solely on mobile devices. The problem of offloading an application to the cloud mainly depends on the following factors: CPU speed of mobile device, network performance, program features, and the efficiency of the cloud server. In consideration of these factors, [3] proposes offloading the whole application to the cloud server without partitioning an application into subcomponents. Although offloading the whole application can usually benefit its execution, not all the components of an application are suitable for being offloaded to the cloud end. For example, the methods of implementing mobile I/O devices and user interfaces should be executed at the mobile end. In addition, some parts of a program, like the ones with light computation requirements but large input data, actually cannot take advantage of remote execution at the cloud server. Application partition technique [4] can support fine-grained offloading, where a mobile application is partitioned into a number of subcomponents, and an optimal decision is made about which components should be offloaded to cloud for computing and which should run locally on mobile devices.

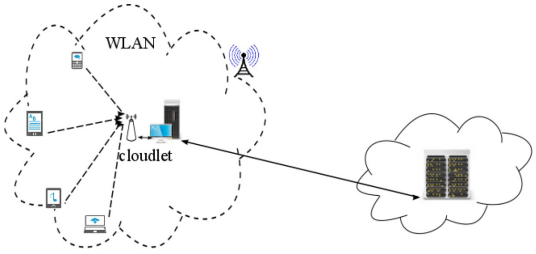


Figure 1 Mobile Cloud System with Cloudlet

For latency-sensitive mobile applications, such as augmented reality with real-time constraints, offloading to the remote cloud is insufficient, because of the high latency of Wide Area Networks (WAN). Cloudlet or Edge cloud is an emerging paradigm designed to better support both latency-sensitive and resource-intensive mobile applications [5]. It is positioned as the middle entity of the three-tier hierarchy: mobile device, cloudlet, and cloud with functionalities of data routing and security guard, similar to a proxy in cloud computing systems. Additionally, cloudlet can speed up mobile application executions by providing powerful computing capabilities. A cloudlet is usually set up at a public place, like a shopping center, theater, office building, or assembly room to enable convenient access for mobile devices.

Resource starvation becomes an inevitable problem in MCC with the exponential increase of mobile services. In addition to traditional aspects of resource

managements, many new challenges beset the problem of resource allocation in MCC systems, such as the application latency, resource demand, computing security, etc. Some investigations have proposed considering the issues in the field of
MCC resource allocation from different perspectives, such as multimedia service, Internet games, system throughput, security, mobile resource [6], etc.

Resource Management Framework

Partition Offloading

One of the main research approaches to MCC systems addresses the efficiency of resource utilization in the partitioning and offloading processes of mobile applications. Such an approach offers to determine which partitions of mobile applications are suitable to be offloaded for remote computing in the cloud. The Branch and Bound (B&B) algorithm based on Linear Programming Solver (LP solver) is being used in schemes proposed in [4] for calculation of offloading decisions concerning application partitions. B&B is a feasible approach for solving integer linear problems when the number of partitions is not large; however, the number of its feasible solutions grows exponentially with the number of partitions, resulting in a high time complexity ($O(2^n)$). That urges us to investigate an algorithm with low computational complexity which can improve model practicability to support the offloading decision calculation in real-time. The strategy of offloading only some parts of mobile applications that can realize benefits from remote execution has been shown to be beneficial and is adopted by MCC [7]. Moreover, the resource allocation of cloud computing for partitioned application further improves the efficiency of cloud resource utilization. Some research, such as in [8], work on the cloud computing resource allocation for mobile requests from application partitions, but few are considering the impacts of the offloading sequence of mobile application partitions to the utilization efficiency of cloud computing resources. We model the resource allocation problem with the considerations of offloading the sequence of application partitions for MCC systems to improve the system capacity.

We proposed a Dynamic Programming based Offloading Algorithm (DPOA) which has low time complexity ($O(n^2)$) in proportion to the square of the number of subprograms in a mobile application [9]. Compared to B&B, DPOA can quickly achieve an optimal offloading strategy, and by shortening the strategy update period, the offloading decision by DPOA can run in real time, which is crucial to the efficiency of real-time applications in MCC systems. We also model the resource allocation problem for partitioned mobile applications as a semi-Markov Decision Process (SMDP) [10]. A system reward model is developed with the consideration of the impacts of allocating computing resources to different partitions, which are classified according to their offloading sequence. The objective is to achieve an optimal allocation policy of cloud computing resources through maximizing the system reward, in order to obtain the maximum system throughput (in terms of request acceptance rate) and to fully utilize the computing resource by preventing tasks being dropped due to resource depletion. Compared with the Greedy approach, our approach not only provides a better allocation policy to speed up the application execution, but also significantly increases the acceptance ratio of service requests from application partitions, especially when system computing resources are limited.

Security Aware Resource Allocation.

Security issues are an inevitable challenge in resource allocation of MCC systems [11]. While some researchers have offered solutions for efficient cloud resource management in MCC systems [12-13], they lack in providing the security guarantee against possible attacks, leading to the loss of the protection ability of cloud. A resource allocation for security services in MCC system has been proposed in [8]. They considered the cloud services composed of two security categories: Critical Security (CS) service and Normal Security (NS) service, as a coarse-grained model. However, the varying resource requirements from mobile users is not considered in their strategy. Moreover, their approach cannot adjust the allocation policy according to the security level of mobile requests and cloud resource availability.

We proposed an adaptive security-aware resource allocation approach that can meet the various resource requirements [13]. The basic idea is to classify the requests from mobile users into multiple risk degree and then consider the resource allocation in order to maximize the overall system benefits. Here risk degree is used to model the security guarantee. For example, a request of low risk degree (meaning low security requirement, e.g., communication can be over public channel and computation can be done without considering privacy) might only need few or no extra resources, while a request of high risk degree demands substantial extra resources (meaning high security requirement, e.g., communication should be over authenticated and confidential channel). Management of resource allocation is modeled as a Semi-Markov Decision Process (SMDP) under an average reward criterion that takes account of the request's risk degree, the current request arrival rate, and the availability of cloud resource. By solving the linear programming problem, our approach can adaptively adjust the resource allocation strategy with the objective of resource protection and throughput maximization.

Multi-resource management for MCC with Cloudlet.

The Cloudlet helps MCC systems meet the requirement of real-time interactive response by means of providing a resource-rich server/cluster in the vicinity of the mobile users, and one-hop high-bandwidth wireless access to the cloudlet. However, the computing resources of the cloudlet are not as rich as the remote cloud cluster, and the wireless bandwidth that connects mobile devices and the cloudlet is limited. There is a high probability that the cloudlet will run out of resources and that no new mobile request can be admitted, if an excessive number of mobile users offload their applications for execution at the cloudlet [14].

We have proposed a joint multi-resource allocation in the MCC system with cloudlet [15]. This system is based on a reward model for resource allocation that takes account of wireless bandwidth, cloudlet and distant cloud computing resources. The allocation scheme considers the system benefits or impacts in accepting or rejecting the new resource request according to the current request traffic, the availability of the system resources, and the QoS guarantee of mobile users. Based on the reward model, a multi-resource allocation strategy is developed, which can adaptively determine whether to accept a new mobile service request for the execution at the cloudlet or the distant cloud. Furthermore, the strategy can determine the optimal amount of wireless bandwidth and computing resources to allocate to the accepted request, and thus achieve the optimal system performance. The SMDP-based multi-resource allocation problem is solved as a linear programming problem using lp solver tool. Extensive performance simulations show that the proposed resource allocation mechanism provides a lower request rejection rate and latency of mobile service compared to those of greedy policies. The proposed multi-resource allocation algorithm can be used in practice by executing the algorithm offline given the various request traffic parameters, the amount of system resource, and the resource price in the reward model. So pre-calculated allocation decisions can be made in the form of a search table for when the request traffic information and availability of system resources are profiled in real-time.

Table 1 SMDP Algorithm

State (s) ={Current #of VM and Bandwidth Utilization for each service class, ***event***}
event ={Arrival from mobile devices, Departure from Cloudlet or Remote Cloud}
Action (a) = {Accept by Cloudlet or Remote Cloud, Reject}
Transition probabilities ={Transition probability from the state ***s*** to the next state under action ***a***}
Reward ={ Lumpsum income – Continuous cost}
Maximize *{*Sum of rewards over all ***s*** *and* ***a****}*
constraints={maximum # of VM and bandwidth resources}

b) Information Management in Dynamic Distributed Databases

The challenges of information management in a *Resilient Edge Cloud Designed Network* derive in part from the dynamic and distributed features of operations in such a network. In particular, to exploit fully computing at the edge it is necessary to take account of the effect of querying on message traffic in the network. The absence of centralized management dictates policies designed to maximize the use of edge nodes which typically have relatively modest storage and processing power. Edge nodes must thus work together in information processing tasks, and this occasions the movement of data between nodes. To minimize the message traffic thus generated requires innovative strategies. One such strategy detailed here is the development of an engineered overlay network structured as a hypercube graph. This strategy was originally proposed in the International Technology Alliance project [16,23]. Research undertaken in that project demonstrated the feasibility of using hypercube routing for query optimization in a Dynamic Distributed Federated Database (the GAIAN DB) built by IBM-UK [16,22]. The current strategy is to implement hypercube routing for query optimization in a software defined network context designed to make the engineered hypercube into an efficient and practical distributed database tool.

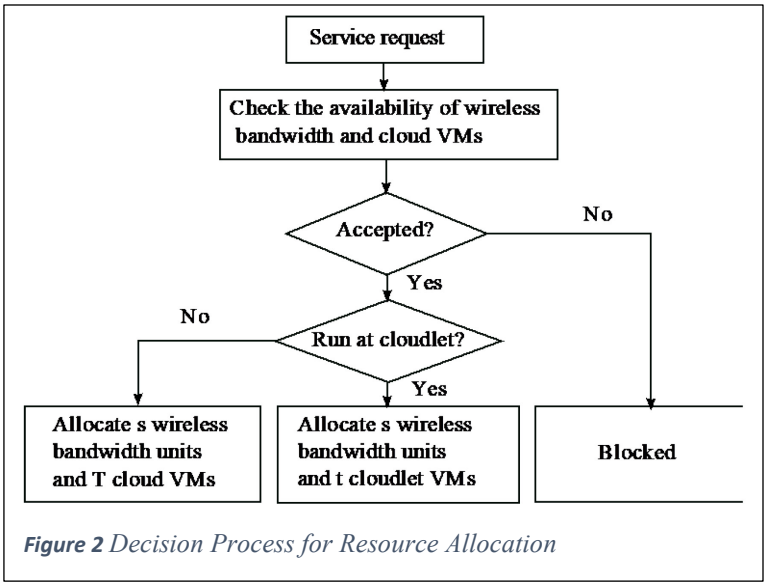


Figure 2 *Decision Process for Resource Allocation*

Slaying the Network Distance Dragon in Query Optimization

Querying in a Dynamic Distributed Federated Database (DDFD) can add significantly to message traffic [16]. Typically, several nodes have information that must be consolidated to satisfy a query. Clearly, data has to be moved over the network to perform operations such as union and join. One critical measure of message traffic is the amount of data (x) to be moved multiplied by the distance (y) moved [17,19]. Thus, message traffic associated with querying can be reduced by minimizing xy for an operation involving several participating nodes in the execution of a query. To do this requires distance information, a requirement that could also add to message traffic if querying is needed to determine distances between nodes. Recognition of this problem has led to the formulation of an engineered DDFD in which inter-node distances can be determined in constant time [5]. "Engineered" means the DDFD is structured as a logical (or virtual) network in the form of an n-dimensional hypercube [21]. The hypercube DDFD is embedded in an underlying physical network (or substrate) such as the Internet. The hypercube has been chosen because the distance between any two nodes in such a graph is just the number of positions in which their respective n-bit labels differ, and determining this Hamming distance is a constant time operation [27].

Hypercubes are desirable for other reasons as well [28]. An n-dimensional hypercube Hn with 2n nodes is regular of degree n. It is robust and resistant to attack as a network structure in that it remains connected with the removal of less than n nodes [18,23a]. Moreover, there are r! distinct paths of length r between nodes that are at distance r from each other, and these paths can be determined from the node labels alone. In addition, the diameter (maximum distance between any two nodes) of Hn is n (= log 2n), and the average length of a path approaches n/2 as n increases. Even with nearly half of the nodes missing, messages can be routed through all the paths of the engineered hypercube [26].

Nothing is truly free in life, and the engineered hypercube network is no exception. At any point in the evolution of a DDFD, an engineered, n-dimensional hypercube structure is not likely to be complete, i.e., not all 2n nodes needed to form the hypercube will be present [26]. This means that some nodes must act for missing ones to ensure the existence of the expected paths in the hypercube. As nodes enter and leave the DDFD, several adjustments must be made in the incomplete hypercube, including assignments for the missing nodes [26]. The operations involved in

maintaining the integrity of the hypercube constitute overhead costs. Studies have shown, however, that under conditions of relatively stability (i.e., the rate at which nodes appear and disappear is modest) the reduction in message traffic given by the engineered hypercube more than offsets the costs of maintaining the structure [26]. Figure 3 shows the tradeoff between the random network (PA) and the engineered hypercube network (HC) for different values of query volume (QC) and rate of network change (RC) [26].

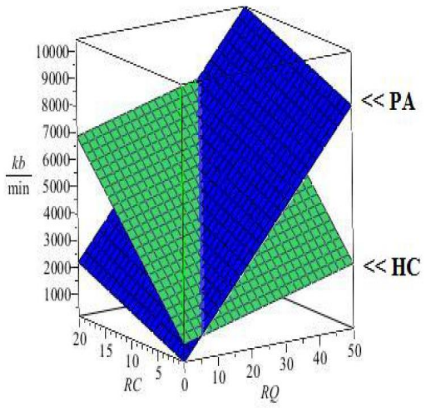


Figure 3 Bandwidth Utilization for Hypercube Networks

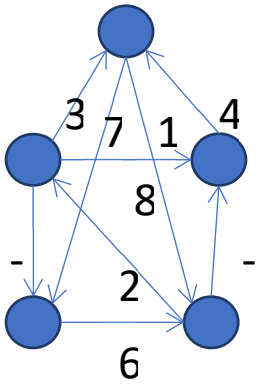
Query Optimization in an Engineered DDFD.

The engineered hypercube DDFD allows for determining inter-node distances cheaply, but query performance is dependent on the number of participating nodes, and the sequence of operations required [17]. Relational joins are particularly demanding. For example, if 3 nodes a,b,c are participating in a join, it is necessary to examine all 3! sequences of the form a ⋈ b ⋈ c (⋈ denoting a join) to determine which one gives the minimum sum of data times distance. Clearly a brute force approach to solving this problem is of exponential complexity. If the number of nodes is small, say at most 6, dynamic programming can provide a real time solution [17]. For larger numbers of nodes, alternatives must be sought.

One alternative is to designate a node that is relatively central to those providing data for the query, and sending all the data to that node to complete the relational operation [24]. The first step in this delegation procedure is finding the central node. The simplest way to do this is by means of the Floyd-Warshall algorithm [29], a dynamic programming approach to finding the shortest paths between every pair of vertices in the subgraph formed by the nodes involved in the query operation. From the matrix of shortest distances produced by this algorithm, a central node can be chosen by comparing the sums of data times distance for each of the nodes in the subgraph. Floyd Warshall executes in O(N3) steps where N is the number of nodes in the subgraph; an additional O(N2) steps is needed to select the central node [29]. The table in Figure 4 gives the results of the Floyd-Warshall algorithm for the weighted graph G.

Assuming the weights shown on the edges of G represent distance times data, the central node is number 4.

If the number of nodes participating in a relational operation is relatively large, say over 1000, a suboptimal consolidation could be achieved efficiently by partitioning the set of nodes. In particular, k-medoids clustering (partitioning around medoids algorithm) could be used to partition the set of nodes into k subsets, for some value of k [30]. This clustering method is related to k-means; it chooses nodes in the given set as cluster centers, and can work with the graph distance measure. The complexity of the computation is of order O(N2). After assigning nodes to clusters, the data in each subset could then be consolidated, and the subset results integrated to yield a response to the query. Such a solution would not necessarily be optimal, but it would substantially reduce message traffic. Additional savings in message traffic can be achieved in executing join operations by first performing semijoins to reduce the amount of



0	1	-3	2	-4
3	0	-4	1	-1
7	4	0	5	3
2	-1	-5	0	-2
8	5	1	6	0

Table of Lengths

***Figure 4** Pairwise Shortest Path Lengths*

data that needs to be transferred to satisfy the query [17].

c) Bio-Inspired Intrusion Detection System (IDS)

Denial-of-service (DoS) or distributed denial-of-service (DDoS) attacks are one type of aggressive and menacing intrusive behavior to online systems such as IoT devices and cloud-based servers [31]. This attack causes severe damage to applications and services running on the victim node, making it difficult for legitimate users to access the service(s) running on that node. On October 2016, site outages involved the targeting of Dyn – a company that controls many of the Domain Name Servers that service American domains. This widely successful attack utilized the now infamous Mirai – a nasty piece of malware that powers an extensive botnet largely populated by IoT devices. This illustrates the urgent need for effective detection of DoS attacks to protect online services.

Even with the use of protective mechanisms like encryption, authentication, and network firewalls, hackers always find ways to compromise these systems and attack the resources and nodes they protect [32]. Also, existing security mechanisms have difficulty in detecting stealthy and zero-day attacks. Hence, an intrusion detection system (IDS) is needed as an added wall of protection.

An intrusion detection system (IDS) is a device or software application that scans a system or network for malicious activities or policy violations and triggers an alert when an incident occurs or logs these malicious activities to a management station. A conventional IDS system uses either a *signature-based* detection technique or an *anomaly based* detection technique. Both techniques have their limitations. Signature-based IDS have low detection rates for *zero-day attacks,* i.e., attacks for which there exists no known signature, while the problem in using an anomaly based IDS is in its high rate of *false positive*s (when normal behavior is flagged as abnormal behavior) [33]. The use of common IDSs for detecting attacks targetting IoT devices, the edge cloud or the cloud infrastructure may not be enough because of the difference in the processing and memory requirements of IoT devices, and cloud-based infrastructure or resource, and the uniqueness of attack surfaces on IoT devices. In addition, the difficulty in finding the exact combination of events that triggered a particular behavior and, more importantly, to label it as malicious is problematic. These problems call for a more sophisticated IDS with an ability to correlate events and accurately differentiate an attack from normal system or network behavior.

The Human body consists of connected cells and tissues and is constantly being attacked by *pathogens*. A pathogen or infectious agent is a biological agent that causes disease or illness to its host. Due to the similarities that exist between the human immune system (HIS) and a distributed system/network like the IoT network/devices [34] [35], the complex activity and procedures followed by the HIS to detect an attack can be abstracted and applied to combat cyber attacks and intrusions that may occur in an IoT, cloud and edge cloud devices and network. To this end, we propose an anomaly-based bio-inspired intrusion detection system (BioIDS) that utilizes intrusion detection approaches followed by the human body to detect cyber-attacks targeting an IoT, cloud or edge cloud devices/network. The artificial immune system (AIS) is a subfield of artificial intelligence and is a class of computationally intelligent systems inspired by the principles and processes of the human immune system. The algorithms that exist in the field of AIS exploit the immune system's characteristics of learning and memory to solve diverse problems. These algorithms are based on HIS models taken from the field of immunology. Two major Immunology models that have been utilized successfully in AIS are the Self-Nonself (SNS) model which leads to the negative selection algorithm (NSA) [33], [36] and the Danger Theory (DT) based model which leads to the dendritic cell algorithm (DCA) [37] [33].

Technical Approach: A bio-inspired technique is proposed to solve the problems of conventional IDS systems used in IoT devices/networks. The models stated in the preceding sections have been applied to the areas of IDS designs. The core of our design depends on the detector generation. The function of the detector is to classify the incoming traffic into normal cells and abnormal cells.

Detector generation:

The first NSA algorithm [36] which was proposed by Forest et. al. is an exhaustive approach. The limitation of this approach is the computational difficulty of generating valid detectors, which grows exponentially with the size of the self [38]. So, to solve the problems of the exhaustive approach, we need a technique for implementing the NSA which locates a detector instead of selecting them at random as in the case of the exhaustive approach. For this, we employ

Algorithm 1: NSA using Genetic Algorithm

Input: S∈U ("self-set"); r_s =self radius
Output: a set of detector D∈U ("detector set")
1: population ← random individuals
2: **for** the specified number of generations **do**
3: **for** the size of the population **do**
4: Select two individuals, (*parent1* and p*arent2*), with uniform probability.
5: Apply crossover with probability C_p to generate two offspring (*child1* and *child2*).
6: Mutate *child1* and *child2* with probability M_p
7: **If** distance (*child1*,*parent1*) > r_s **and** fitness (*child1*) > fitness (*parent1*) **then**
8: *parent1* ← *child1*
9: **end if**
10: **If** distance (*child2*,*parent2*) > r_s **and**

an evolutionary approach using a genetic algorithm (GA).

GAs are adaptive, heuristic search algorithms based on the evolutionary ideas of natural selection and genetics. As such they represent an intelligent exploitation of a random search used to solve optimization problems. Each generation consists of a population of character strings that are analogous to the chromosome that we see in our DNA. Each individual represents a point in a search space and a possible solution. The individuals in the population are then made to go through a process of evolution [39]. Algorithm 1, shows the steps taken by our approach to detector generation using GA.

A detector is defined as d = (c, r_d), where c = (c_1, c_2, …, c_m) is an m-dimensional point that corresponds to the center of a unit hypersphere with r_d as its radius. In this detector generation phase, the main task is to generate a set of detectors, with the center of each detector being at least (r_d + r_s) distance away from the center of its nearest self element (which has raduis = r_s).

The GA is initialized with random individuals. Two parents; P_1 and P_2 are drawn from the initialized samples at random with equal probability. Crossover is performed with these two parents to create two offspring (C_1 and C_2) with a probability C_p called the crossover probability. The kind of crossover used here is the one-point crossover.

To perform mutation on C_1 and C_2, a position in the chromosome of each is chosen at random from [1,2…m] and flipped with a mutation probability M_p. This flipping is done by replacing the value of the attribute in the randomly selected location with a randomly generated value that lies between [0, 1].

After mutation, the fitness of C_1 and C_2 is evaluated using the function in (2):

$$fitness(individual) = e^{-r_s/D} \qquad (2)$$

Where r_s is a threshold value (allowable variation) of a self point; in other words, a point at a distance greater than r_s from the self sample is considered to be abnormal. D is the distance between the individual and the nearest self. This distance measure is calculated using the Euclidean distance measure given by (3):

$$D(X,Y) = \left(\sum_{i=1}^{n}(x_i - y_i)^2\right)^{1/2} \qquad (3)$$

The fitness of both parents used for the crossover together with the resulting offspring is calculated. Also, the distances between (C_1 and P_1) and (C_2 and P_2) are both calculated. P_1 in the population is replaced by C_1 if C_1 has a better fitness and the distance between its center and that of P_1 is greater than r_s. Similar action is taken for replacing P_2. The fittest individuals are selected as the detectors.

IDSs that were designed based only on the self-nonself model [37] were prone to high false positive rates. The DT model was adopted to address this issue. Some researchers have proposed IDS that utilizes the model for intrusion detection. Given that zero-day attacks and high rates of false positives are the problems we would want to solve in conventional IoT intrusion detection systems, our proposed IDS for IoT devices/networks would combine the strength of both models to form a more practical and efficient IDS for IoT devices/networks. To fully integrate both models to create an efficient IDS, Figure 5 shows the block diagram of the proposed system to classify an activity as an attack. Using these five modules of Figure 5, the detection process can be divided into four major steps.

In Step 1, basic features are generated or extracted from ingress network traffic to the internal network where protected servers reside and are used to form traffic records for a well-defined time interval. In Step 2, the signal selection process is performed. Signals that represent abnormal activities (danger signals) and normal activities (safe signals) are generated using the features extracted in Step 1. Step 3 involves two phases (i.e., the training phase and the testing phase). In the training phase, the NSA instances that make up the NSA module are trained using the two signals generated in Step 2. Our system uses a weighted majority vote technique. Hence, in Step 3, as part of the training phase, the weights to be assigned to all NSA and DCA instances are computed using the training data. In the test phase, profiles for individual observed traffic records are built. Then, the tested profiles are handed over to the NSA and DCA modules respectively. The NSA module compares the individual tested profiles with the stored detectors. The DCA samples the individual tested profiles using a sampling technique. The decisions made by the constituent NSA/DCA modules are weighted (using the weights determined during the training phase) and used by the *decision-making* module to distinguish DoS attacks from legitimate traffic.

Evaluation and Experimentation

The development and evaluation of bio-inspired intrusion detection system are performed according to the following:

(Analysis I): The first plan for our work involves performing detailed analyses of IoT device/network attack surfaces and how inherent vulnerabilities can be exploited by malicious users.

(Analysis II and Experimentation I): The second step involves defining features that would identify an entity (an activity, or process). Using the knowledge gained in *(Analysis I)*, we carefully select features that will be used to signify dangerous and safe signals respectively. This also involves developing a lightweight software agent that will run on IoT devices/edge-cloud nodes, especially on those that have constraints on memory and processing capability. These agents will be tasked with the responsibility of gathering data which will be sent to the nearest edge cloud-based node running the full version of out proposed IDS (shown in Figure 5). This data is used in detecting both network level and device level intrusions. Figure 6 shows how the agent could be

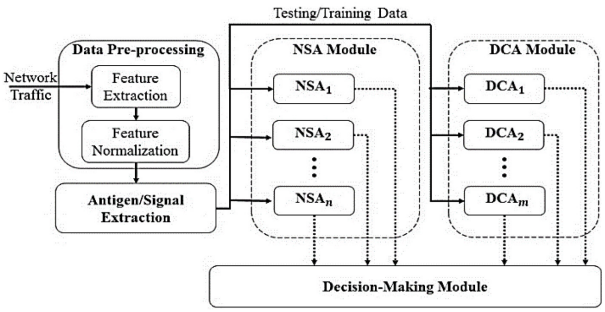


Figure 5 *Bio-Inspired intrusion detection system (BioIDS) building blocks*

distributed in the IoT network to aid in intrusion detection.

(Experimentation II and Evaluation): To evaluate the effectiveness of our solution, we use the following two approaches. One approach will be to simulate the proposed IDS design and architecture using the integrated testbed prosed under this project. The second approach involves setting up a private cloud under the Cyber Defense Technology Experimental Research Laboratory (DeterLab) sponsored by the Department of Homeland Security (DHS), and using the low capacity nodes under this testbed as the IoT nodes.

Once the testbed is set up, we will generate normal network traffic for 3 months. This normal traffic will be used to train the NSA instances.

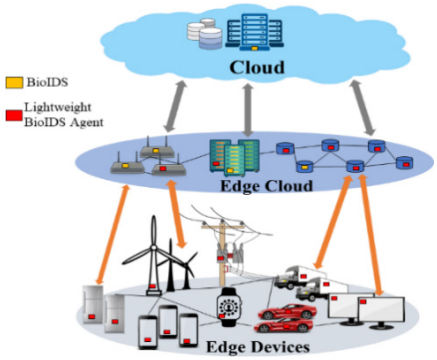


Figure 6 *Architecture for IoT device/network security using BioIDS and lightweight*

To evaluate our model's effectiveness, we will perform penetration testing in real-time against the IoT devices or cloud/edge-cloud nodes using the information and vulnerability analysis results obtained from Analysis I. Our system is being assessed based on its false positive rate (FPR), and Detection Rate (DR). A good result will be one with very high DR and very small FPR. To achieve this we are currently establishing, through Intrnet2 and other networks, a direct connection between the Kyushu Institute of

Technology in Japan and our lab at CCNY in USA to experiment with our model in real-time traffic. We plan also to adapt our systems to the virtual machine environment as highlighted in [40] [41].

Conclusion

This paper addressed three main topics for secure edge could network design; 1) resource management in mobile cloud computing; 2) information management in dynamic distributed databases; 3) biological-inspired intrusion detection system. Currently our team at the City University of New York, City College (CCNY) and our partner at Kyushu Institute of Technology (Kyutech), plan to continue further investigation of secure edge cloud network and to measure in real time, through the direct connection between the labs at Kyutech and CCNY, the various system parameters under investigation.

References

[1] Z. Zhou and D. Huang," Efficient and secure data storage operations for mobile cloud computing," Proc. of the 8th International Conference on Network and Service Management, CNSM '12, pp. 37-45, Laxenburg, Austria, 2013.

[2] K. Yang, S. Ou, and H. Chen, "On Effective offloading services for resource- constrained mobile devices running heavier mobile internet applications," Comm. Mag., 46(1):56-63, January 2008

[3] K. Kumar and Y. Lu, "Cloud computing for mobile users: Can offloading computation save energy?," Computer, 43(4):51-56, April 2010

[4] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making smartphones last longer with code offload," Proc. of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys '10, pp. 49-62, New York, 2010

[5] M. Satyanarayanan, "The role of cloudlets in hostile environments," Proc. of the Fourth ACM Workshop on Mobile Cloud Computing and Services, MCS '13, pp. 1-2, New York, NY, 2013

[6] S. Wang and S. Dey, "Adaptive mobile cloud computing to enable rich mobile multimedia applications," IEEE Transactions on Multimedia, 15(4):870-883,2013

[7] A. Ellouze, M. Gagnaire, and A. Haddad, "A mobile application offloading algorithm for mobile cloud computing," Prof. of Third IEEE Mobile Cloud Computing, Services and Engineering, Mobile Cloud 2015, San Francisco, CA, March 2015

[8] D. Hoang, D. Niyato, and P. Wang, "Optimal admission control policy for mobile cloud computing hotspot with cloudlet. Prof. of IEEE Wireless Communications and Networking Conference, WCNC 2012, pp. 3145-3149, Paris, France, April 2012

[9] Y. Liu and M. J. Lee, "An effective dynamic programming offloading algorithm in mobile cloud computing system," Prof. of IEEE Wireless Communications and Networking Conference, WCNC 2014, pp. 1868-1873, Istanbul, Turkey, April, 2014

[10] Y. Liu and M. J. Lee, "An adaptive resource allocation algorithm for partitioned services in mobile cloud computing," Prof. of Ninth IEEE International Symposium on Service-Oriented System Engineering, SOSE 2015, San Francisco, CA, March 2015

[11] S. Malik and M. M Chaturvedi, "Privacy and security in mobile cloud computing: Review," International Journal of Computer Applications, 80(11):20-26, October 2013

[12] Q. Xia, W. Liang, and W. Xu, "Throughput maximization for online request admissions in mobile cloudlets," Prof. of 38th Annual IEEE Conference on Local Computer Networks, pp. 589-596, Sydney, Australia,, October, 2013

[13] Y. Liu and M. J. Lee, "Security-aware resource allocation for mobile cloud computing systems, " Prof. of 24th International Conference on Computer Communication and Networks, ICCCN 2015, Las Vegas, NV, USA, 2015

[14] S. Clinch, J. Harkes, A. Friday, N. Davies, and M. Satyanarayanan, "How close is close enough? Understanding the role of cloudlets in supporting display appropriation by mobile users," Prof. Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on, pages 122 -127. 2012

[15] Y. Liu and M. Lee, Yangyang Zheng, "Multi-Resource Allocation for Cloudlet-Based Mobile Cloud Computing System," *IEEE Transaction on Mobile Computing*, Vol. 15, No. 10, Oct. 2016

[16] G. Bent, P. Dantressangle, D. Vyvyan, A. Mowshowitz, and V. Mitsou, "A dynamic distributed federated database," Annual Conference of the International Technology Alliance, Imperial College-London, 2008.

[17] A. Kawaguchi, A. Mowshowitz, A. Nagel, A. Toce, G. Bent, P. Stone, and P. Dantressangle, "A model of query performance in Dynamic Distributed Federated Databases taking account of network topology," Annual Conference of the International Technology Alliance, Southampton, UK, September 2012.

[18] A. Mowshowitz, V. Mitsou, and G. Bent, "Evolving networks and their vulnerabilities," In: Dehmer, M., et al. editors, Modern and Interdisciplinary Problems in Network Science: A Translational Research Perspective, CRC Press, 2018.

[19] A. Mowshowitz, A. Kawaguchi, A. Toce, A. Nagel, P. Stone, P. Dantressangle and G. Bent, "Network topology as a cost factor in query optimization," Annual Conference of the International Technology Alliance, Adelphi, MD, 2011.

[20] A. Mowshowitz, A. Kawaguchi, A. Toce, A. Nagel, G. Bent, P. Stone, and Dantressangle, "Query optimization in a distributed hypercube database," Annual Conference of the International Technology Alliance, Imperial College-London, September 2010.

[21] A. Mowshowitz, G. Bent, and P. Dantressangle, "Aligning network structures: embedding a logical dynamic distributed database in a MANET," Annual Conference of the International Technology Alliance, University of Maryland, 2009.

[22] A. Mowshowitz, V. Mitsou, and G. Bent, "Models of network growth by combination," Annual Conference of the International Technology Alliance, Imperial College-London, 2008.

[23] A. Mowshowitz, and G. Bent, "Formal properties of distributed database networks," Annual Conference of the International Technology Alliance, University of Maryland, 2007.

[23a] P. Stone, P. Dantressangle, G. Bent, A. Mowshowitz, A. Toce, and B. Szymanski, "Query execution and maintenance costs in a Dynamic Distributed Federated Database," Annual Conference of the International Technology Alliance, Southampton, UK, September 2012.

[24] P. Stone, A. Mowshowitz, P. Dantressangle, and G. Bent, "Calculation of the center-of-data in a hypercube," Annual Conference of the International Technology Alliance, Southampton, UK, September 2012.

[25] P. Stone, P. Dantressangle, A. Mowshowitz and G. Bent, "Review of relational algebra for Dynamic Distributed Federated Databases," Annual Conference of the International Technology Alliance, Imperial College-London, September 2010.

[26] A. Toce, A. Mowshowitz, A. Kawaguchi, P. Stone, P. Dantressangle, and G. Bent, "An efficient hypercube labeling scheme for dynamic Peer-to-Peer networks," J. Parallel Distrib. Comput. 102, 2017, pp. 186-198.

[27] A. Toce, A. Mowshowitz, A. Kawaguchi, P. Stone, P. Dantressangle, and G. Bent, "HyperD: Analysis and performance evaluation of a distributed hypercube database," Annual Conference of the International Technology Alliance, Southampton, UK, September 2012.

[28] Wikipedia. Hypercube Graphs. https://en.wikipedia.org/wiki/Hypercube_graph

[29] Wikipedia. Floyd-Warshall Algorithm. https://en.wikipedia.org/wiki/Floyd–Warshall_algorithm

[30] Wikipedia. k-medoids. https://en.wikipedia.org/wiki/K-medoids

[31] Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu, "A system for denial-of-service attack detection based on multivariate correlation analysis," *IEEE transactions on parallel and distributed systems*, vol. 25, no. 2, pp.447-456, 2014.

[32] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems," *Journal of network and computer applications*, vol. 30, no. 1, pp.114-132, 2007.

[33 a] O. Igbe, O. Ajayi, and T. Saadawi, "Detecting Denial of Service Attacks using a Combination of Dendritic Cell Algorithm and the Negative Selection Algorithm," in *the 2nd IEEE International Conf. on Smart Cloud 2017*, November 2017.

[33 b] O. Igbe, O. Ajayi, and T. Saadawi, "Denial of Service Attack Detection using Dendritic Cell Algorithm," in *the IEEE 8th Annual Ubiquitous Computing, Electronics, and Mobile Communication Conference (UEMCON)*, October 2017.

[33 c] O. Igbe, I. Darwish, and T. Saadawi,"D eterministic Dendritic Cell Algorithm application to Smart-Grid Cyber Attack Detection," in *the IEEE 4th International Conference on Cyber Security and Cloud Computing*, June 2017.

[34] A. Somayaji, S. Hofmeyr, and S. Forrest, "Principles of a computer immune system," in *Proceedings of the 1997 workshop on New security paradigms*. ACM, 1998, pp.75-82.

[35] M. Zamani, M. Movahedi, M. Ebadzadeh, and H. Pedram, "A ddosaware ids model based on danger theory and mobile agents," in *Computational Intelligence and Security, 2009. CIS'09. International Conference on*, vol. 1. IEEE, 2009, pp.516-520.

[36] S. A. Hofmeyr and S. Forrest, "Architecture for an artificial immune system," *Evolutionary computation*, vol. 8, no. 4, pp. 443-473, 2000.

[37] F. Gu, J. Greensmith, and U. Aicklein, "The dendritic cell algorithm for intrusion detection," *Biologically Inspired Networking and Sensing: Algorithms and Architectures*, pp.84-102, 2011.

[38] D. Dasgupta, F. Nino, "Immunological computation: theory and applications," Auerbach Publications, 2008.

[39]Introduction to Genetic Algorithms, http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/hmw/article1.html (accessed January 2, 2016).

[40] K. Kourai, K. Nakamura, "Efficient VM Introspection in KVM and Performance Comparison with Xen", Department of Creative Informatics, Kyushu Institute of Technology, Fukuoka, Japan, November 2014

[41] K. Kourai, K. Juda, "Secure Offloading of Legacy IDSes Using Remote VM Introspection in Semi-trusted Clouds", Department of Creative Informatics, Kyushu Institute of Technology, Fukuoka, Japan, June 2016