Adaptive Alert Management for Balancing Optimal Performance among Distributed CSOCs using Reinforcement Learning

Ankit Shah, Rajesh Ganesan, Sushil Jajodia, *Fellow, IEEE,* Pierangela Samarati, *Fellow, IEEE,*
and Hasan Cam, *Senior Member, IEEE*

Abstract—Large organizations typically have Cybersecurity Operations Centers (CSOCs) distributed at multiple locations that are independently managed, and they have their own cybersecurity analyst workforce. Under normal operating conditions, the CSOC locations are ideally staffed such that the alerts generated from the sensors in a work-shift are thoroughly investigated by the scheduled analysts in a timely manner. Unfortunately, when adverse events such as increase in alert arrival rates or alert investigation rates occur, alerts have to wait for a longer duration for analyst investigation, which poses a direct risk to organizations. Hence, our research objective is to mitigate the impact of the adverse events by dynamically and autonomously re-allocating alerts to other location(s) such that the performances of all the CSOC locations remain balanced. This is achieved through the development of a novel centralized adaptive decision support system whose task is to re-allocate alerts from the affected locations to other locations. This re-allocation decision is non-trivial because the following must be determined: (1) timing of a re-allocation decision, (2) number of alerts to be re-allocated, and (3) selection of the locations to which the alerts must be distributed. The centralized decision-maker (henceforth referred to as agent) continuously monitors and controls the level of operational effectiveness-LOE (a quantified performance metric) of all the locations. The agent's decision-making framework is based on the principles of stochastic dynamic programming and is solved using reinforcement learning (RL). In the experiments, the RL approach is compared with both rule-based and load balancing strategies. By simulating real-world scenarios, learning the best decisions for the agent, and applying the decisions on sample realizations of the CSOC's daily operation, the results show that the RL agent outperforms both approaches by generating (near-) optimal decisions that maintain a balanced LOE among the CSOC locations. Furthermore, the scalability experiments highlight the practicality of adapting the method to a large number of CSOC locations.

Index Terms—Distributed cybersecurity operations center (CSOC), centralized alert management, level of operational effectiveness, reinforcement learning, and adaptive resource allocation.

1 Introduction

In this era of digitization, where the critical data of an organization are transmitted and stored electronically, monitoring and protecting computer networks have become more critical than ever. Organizations rely on cybersecurity operations center (CSOC), a unique amalgamation of people, processes, and technology, to protect against the ever-increasing cybersecurity threats. Strategically placed sensors monitor the traffic flow in the network, and sensor data are analyzed by automatic processing units such as intrusion detection systems (IDSs) and secure information and event management (SIEM) tools for malicious threats. Alerts are generated when suspicious activities are detected, which in turn require cybersecurity analysts to analyze them thoroughly. Analysts, working in shifts, help in detecting, analyzing, and reporting significant alerts and thereby help in preventing or limiting the impact of cybersecurity incidents. The alert analysis process is shown in Figure 1.

Large organizations typically have CSOCs distributed at multiple locations. Each CSOC location is independently managed by hiring and scheduling analyst resources such that the alert workload generated by the monitored sensors is investigated in a timely manner and the quality of analysis is maintained by having an ideal mix of analyst expertise levels (skills) in a shift [1]. The performance of a CSOC is dependent on the *timely identification of significant alerts*, which is the focus of this paper. In Shah et al. [2], the authors propose a novel metric to quantify the performance of a CSOC. The performance, defined as the level of operational effectiveness (LOE), of a CSOC is measured by calculating the average total time taken for alert investigation per hour (avgTTA/hr). The avgTTA/hr metric is the average of the sum of waiting time in the queue and the investigation time of all the alerts investigated in that hour. Under normal operating conditions, a baseline avgTTA/hr value is established that is acceptable to the CSOC along with a threshold value for avgTTA/hr. The LOE is continuously monitored using a color-coded representation as shown in Figure 2, where different tolerance bands (colored-zones) are created based on the baseline and threshold values of avgTTA/hr. In this work, the LOE metric (avgTTA/hr) and the color-coded

A. Shah, R. Ganesan and S. Jajodia are with the Center for Secure Information Systems, George Mason University, Fairfax, VA 20030 USA e-mail: {ashah20, rganesan, jajodia}@gmu.edu.
P. Samarati is with the Computer Science Department, Università degli Studi di Milano, 26013 Crema-Italy. email: pierangela.samarati@unimi.it.
H. Cam is with the U.S. Army Research Laboratory, Adelphi, MD 20783 USA e-mail: hasan.cam.civ@mail.mil.
Shah, Ganesan, and Jajodia were partially supported by the Army Research Office under grants W911NF-13-1-0421 and W911NF-15-1-0576 and by the Office of Naval Research under grant N00014-15-1-2007.
Manuscript received xxxx 1xx, 2018; revised xxxx xx, 2018.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

monitoring framework are used to measure the performance of the individual CSOC locations.

There are organizational and adversarial events that affect the performance of a CSOC. Examples of adverse events and their impact on a CSOC include: (1) increase in intensity and duration of surges in alert generation due to an attack or restoration of a broken communication link between the sensor/IDS and CSOC, and (2) increase in alert investigation time due to new vulnerability discoveries or analyst absenteeism. As a result, the avgTTA/hr increases and the LOE of a CSOC decreases. Under the influence of an adverse event, a CSOC can marginally improve the LOE by increasing the alert investigation rate by canceling/delaying the non-alert analysis related tasks of the available analysts and by calling in managers/supervisors to assist with the alert analysis work. However, to further increase the throughput of the alert analysis process for timely identification of significant alerts, a CSOC has to re-allocate alerts to other CSOC location(s) or rely on external analyst resources. In Shah et al. [3], the authors propose a dynamic decision-making framework by allocating an on-call analyst workforce to control the LOE of a CSOC. An on-call resource allocation policy is provided in [3] with limited number of additional resources available in a given time-period. The study focused on a single CSOC location and did not take into account multiple CSOC locations with varying performance requirements. It is to be noted that summoning an on-call analyst workforce is expensive. Hence, when faced with an adverse event, it is beneficial for a CSOC to re-allocate alert workload among the other locations for analysis. Due to the uncertainties arising from adverse conditions, alert generation rates, and threat indicators of the alerts that impact the alert analysis process at the CSOC locations, decision-making to reallocate is non-trivial. For example, re-allocating alert workload from the impacted location to another location whose LOE is ideal at current time, t, can be impacted by an adverse event between time t and $t+1$. As a result, the LOE at the latter location will decrease significantly with the additional alert workload. The desideratum of large organizations and the research objective of the paper are to have uniformly balanced individual performances (LOE) among the CSOC locations in every shift of operation. Hence, the following must be determined for the centralized sequential decision-making under uncertainty: (1) timing of the re-allocation decision, (2) number of alerts to be re-allocated, and (3) selection of the location(s) to which the alerts must be distributed such that LOE of all the CSOC locations remain uniformly balanced.

In this paper, an intelligent decision-making tool is developed using a learning-based optimization framework to determine when, how much, and whom to re-allocate the alert workload to, such that the LOE of all the CSOC locations remain uniformly balanced in the face of adverse events. The proposed framework is based on the principles of stochastic dynamic programming and is solved using reinforcement learning (RL) [4], [5]. The RL-based optimization model continuously monitors the avgTTA/hr metric (backlog of alerts) of all the locations while making re-allocation decisions to balance LOE among the locations. Several real-world scenarios are simulated to learn the long-run values of the states of the system. The goal is to move from one good state to another after learning the long-run values of the system states under a real-world realization of uncertainty (adverse events). Through simulated experiments, the RL-based approach is subjected to several sample realizations of CSOC operations, and the results are compared with both rule-based and load balancing approaches (explained in Section 4) that were devised through conversations with several CSOC managers. The results show that the RL-based approach outperforms both rule-based and load balancing approaches in 1) maintaining and uniformly balancing the individual performances (LOE) between the CSOC locations by selecting (near-) optimal alert workload to re-allocate among the locations, and 2) determining the (near-) optimal timings to implement the decisions. In particular, the results show that the RL-based approach learns to make timely adjustments to the LOE of the CSOC locations by allocating resources that keep the system in good states (where individual LOE are uniformly balanced among the locations) in the long run. As a result, re-allocation decisions are made more often but in smaller proportions (smaller number of resources allocated) compared to the reactive approaches (rule-based and load balancing), which wait for the LOE to significantly deteriorate before making a sub-optimal re-allocation decision.

The contributions of the paper are as follows.

1) The primary contribution is a novel adaptive alert management framework that amalgamates performance metrics, analyst investigation rate, and alert generation rates from sensors of all the CSOC locations within an optimizing simulator for adaptive alert management.
2) A sequential decision-making tool using this framework is presented, which centrally manages the performances (by uniformly balancing the LOE) of all the CSOC locations of an organization by selecting the right size of alert workload and re-allocates it among various locations under the influence of adverse events.
3) Also, more profound insights into the decisions made by the RL-based optimization strategy are provided by comparing scenarios with rule-based and load balancing approaches.
4) Finally, the scalability experiments highlight the practicality of adapting the method to a large number of CSOC locations.

The paper is organized as follows. Section 2 presents the related literature. Section 3 presents the adaptive alert management model; the stochastic dynamic programming formulation and algorithm are described here. Section 4 presents the experimental set-up, and a description of the rule-based and load balancing strategies used for comparison with the developed RL model. Section 5 presents analysis of the experiment results and discusses the computational complexity of the problem. Section 6 presents the conclusion and directions for future research.

2 Related Literature

A CSOC protects an organization from cybersecurity threats using a combination of technological elements and skilled

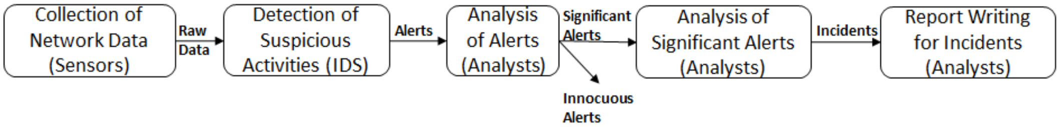

Collection of Network Data (Sensors)
Raw Data
Detection of Suspicious Activities (IDS)
Alerts
Analysis of Alerts (Analysts)
Significant Alerts
Innocuous Alerts
Analysis of Significant Alerts (Analysts)
Incidents
Report Writing for Incidents (Analysts)

Fig. 1. Alert Analysis Process

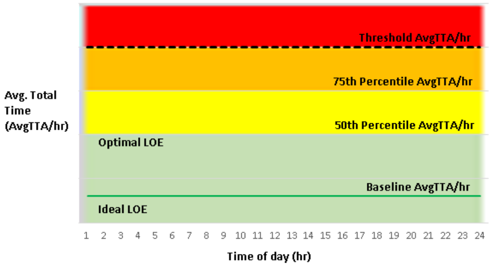


Fig. 2. Color-coded Level of Operational Effectiveness Status [2]

personnel. A CSOC provides many services to an organization, which are broadly categorized into reactive services, proactive services, and security quality management services [6]. In a study conducted by Killcrece et al. [7] at Carnegie Mellon University, alert management related services were among the major services offered by the surveyed security centers. In the alert management service provided by a CSOC, raw data (network traffic data or host-based log data) is collected and monitored by strategically placed sensors in an organization's network. The data is then filtered through an IDS such as SNORT or a SIEM tool (ArcSight [8]). Using pattern matching [9], [10] and/or anomaly based techniques [11], suspicious activities (alerts) are identified from the raw data. Many of the suspicious activities generated by the automated detectors are innocuous. Automating the detection of malicious threats [12], [13] and reduction of innocuous suspicious activities [14], [15] are active areas of research in the field of intrusion detection.

The suspicious activities, identified as alerts, by the automated detectors are further analyzed by the cybersecurity analysts, who are also known as Tier 1 analysts [16], first responders, or real-time analysts [17]. Identification of innocuous and significant alerts is a relatively fast decision made by the analysts [16], [17]. Analysts are the most critical components of a CSOC, as their decisions have a direct influence on the operational efficiency [18]. Recent work in cybersecurity literature has focused on the various roles performed by the analysts in creating security situational awareness at the CSOCs (through a cognitive task analysis study [17]), on understanding the burnout of the analysts (through an anthropological study [18]), and in increasing the efficiency of analysts [19].

Previous work has also addressed various measures of performance and effectiveness of CSOCs. In the pioneering work by Ganesan et al. [1], the authors present a methodology for obtaining an optimal schedule for analysts with respect to minimizing the number of unanalyzed alerts (performance metric) that remain at the end of each shift. Furthermore, scheduling of on-call analysts under uncertainty with respect to the above performance metric of unanalyzed alerts is studied using reinforcement learning in Ganesan et al. [20] and a two-stage stochastic model in Altner et al. [21]. In an anthropological study conducted by Sundaramurthy et al. [18], the average time taken to analyze a significant alert by analysts is one of the operational performance metrics used by the surveyed security centers. In Newcomb et al. [22], the authors present an alert prioritization framework for high value assets in an organization by minimizing the waiting time for the investigation of alerts related to such assets. The quality metric of how accurately the malicious threats are detected is measured by the recall and precision values [23]. The quality of alert analysis process (with low false positive and false negative rates) is maintained by scheduling an ideal expertise mix of analysts (for example junior, intermediate, and senior) per shift [1], [20], [21]. Timeliness is an important security metric [23], which in the context of a CSOC measures the ability to timely identify the significant alerts and take further actions. In Shah et al. [2], the authors quantify the effectiveness of a CSOC by measuring the average time spent by the alerts (sum of the waiting time in the queue and investigation time by the analysts) in the CSOC system per hour (avgTTA/hr). In this work, the above metric, avgTTA/hr, is used to measure the effectiveness of the various CSOC locations.

To attain a balanced operational effectiveness across all locations for the entire time horizon requires decision-making to re-allocate alert workload at each epoch. This sequential decision-making also must consider the future uncertainties in the form of adverse events that may occur at the CSOC locations. Due to the stochastic nature of the events, this decision-making is non-trivial. In discussions with several CSOCs, we found that the current methods used by the CSOC operators for alert and resource management are myopic, which do not take future uncertainties into consideration, and are often rule-based. The current rule-based techniques employed by the CSOCs are used for comparison in the work by Shah et al. [3], which was in collaboration with the Army Research Laboratory. Introduced by Bellman [24], stochastic dynamic programming is widely used for modeling and solving problems of sequential decision-making under uncertainty. Problems are formulated as Markov decision processes [25]. Reinforcement Learning (RL) [4] is one of the ways to solve them. RL models are used in diverse applications in literature to make (near-)optimal decisions. To utilize idle resources during under-utilization and to avoid high response times during over-utilization of resources in computing cloud environments, Yazir et al. [26] propose a dynamic resource allocation model using distributed multiple criteria decision analysis. Duggan et al. [27] propose a RL-based decision support system that schedules virtual machine migrations at optimal times in the wake of network saturation at data centers. In [28], the authors propose a multi-agent RL method for job scheduling (load balancing) in Grid computing.

Farahnakian et al. [29] propose a RL-based dynamic consolidation method to minimize the number of active hosts in cloud data centers to reduce energy consumption. Shaw et al. [30] propose an advanced RL consolidation agent to optimize the distribution of virtual machines (VM) in a data center to improve energy consumption. In this study, the authors evaluate the performance of their approach against a rule-based heuristic that selects from a decreasing order of the most energy efficient hosts for VM allocation. The above two studies use Q-learning to learn the optimal policies of actions, in which the RL algorithms objective is to find (near-) optimal values of expected rewards of the state-action pairs. In contrast, the study presented in this paper employs a stochastic dynamic programming framework to approximate the value of being in a particular state. It is to be noted that the convergence of the RL algorithm will be slower (requiring many iterations) in the former method.

Dynamic load balancing has been a topic of interest for researchers since many decades in various domains [31], [32]. There are many methods used for load balancing in a dynamic environment in literature such as genetic algorithm [33], ant colony optimization [34], active clustering [35], and weighted least connection (WLC) and exponential smooth forecast based on WLC (ESWLC) [36]. All of the above mentioned heuristic methods are essentially static (not sequential) decision-making within their applications, and are reactive (do not consider future uncertainties) to the observed state of the system at the time of decision making. Also, some of the above methods are specific to a particular domain. They appear to be dynamic because the reactive decisions can be taken as often as needed, and the decisions are independent of each other. For example, WLC and ESWLC are used for dynamic load balancing for cloud computing. While WLC does not taken into consideration the resource capacity at each node, ESWLC predicts which node is to be selected based on an exponential smoothing.

Dynamic algorithms need to monitor various attributes of a system such as current load (demand), future (estimated) load, identification of the nodes to which the load need to be transferred to, and throughput of each node. One of the major issues of a complex centralized system is the overload of information. In such systems, capturing the relevant information in a compact manner and decision-making for load balancing is non-trivial.

The work presented in this paper differs from the above as per the following:

1) A sequential decision-making model under uncertainty has not been studied in literature for optimally balancing the individual CSOC performances by re-allocating the alert workload among the CSOC locations.
2) The paper differs from traditional load or line balancing approaches because of the cyber security context, which has uncertainties arising from adverse conditions, alert generation rates, and threat indicators of the alerts that impact the alert analysis process at the CSOC locations. Thus, decision-making to reallocate for balancing LOE is different from load or line balancing that typically has deterministic quantities to allocate.
3) This work presents a novel way of observing and capturing the state of the system by aggregating information from all of the nodes (locations) to reduce the information overload faced by a centralized agent.
4) A first of its kind RL-based approach using stochastic dynamic programming is presented in this paper that finds (near-) optimal long-term rewards for all possible states in contrast to all possible state-action pairs found in recent studies [29], [30], which also used RL approach for load balancing, and thereby achieves a faster convergence during learning due to lower dimensional information requirement (states versus state-action pairs).

The framework for the adaptive model for alert management at the CSOC locations is presented next.

3 Adaptive Alert Management Model Framework

In this section, we present the framework for adaptive alert management model.

3.1 Problem Definition

The research objective is to build a centralized dynamic decision making system that will adaptively and autonomously make the reallocation of alerts decision (when, how much, and to whom) in order to uniformly balance the individual performances (LOE) between the CSOC locations. The avgTTA/hr value, measured in terms of the number of alerts that are backlogged at a CSOC location, must be minimized and balanced among the CSOC locations. Define $X_{i,t}$ as the number of alerts that are backlogged at CSOC location i at time t. The objective is to make decisions such that

$$Min \ \sum_{i=1}^{n} X_{i,t}, \ \ \forall t, \tag{1}$$

and

$$Min \ \sum_{i=1}^{n-1} \sum_{k=i}^{n-1} (|X_{i,t} - X_{k+1,t}|) \ \ \forall t \tag{2}$$

among n locations for each hour t in a day of operation, and over any given work cycle. The index $k+1$ in the above equation represents another CSOC location. Typically, a CSOC follows 14-day shift schedule for the analysts [1], [21]. Equation (1) ensures that the total backlog is minimized across all locations, and Equation (2) ensures that the absolute difference between the backlogs between any two locations is minimized (uniformly balanced backlog among all locations). The two equations are combined into a single contribution function as explained later. Figure 3 shows the model framework, which consists of (1) a simulation model and (2) an optimization model. Alert analysis processes at various CSOC locations are simulated and, using a reinforcement learning-based optimization model,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

re-allocation decisions for the alert workload are made to meet the research objective stated above. The details of the simulation and optimization models are explained next.

3.2 Simulation Model

Multiple CSOC locations are simulated. Each CSOC location has enough number of analysts per work-shift for an expected alert workload generated from the monitored sensors such that an acceptable avgTTA/hr value (LOE metric) is maintained. The queueing system is modeled as M/D/c/FCFS, which combines the service rates of all analysts into a single server and calculates the average alert arrival rate per hour for the CSOC location by combining the alert arrival rates of all the sensors [2]. Based on predetermined threshold values for avgTTA/hr, different color-coded tolerance bands (Figure 2 [2]) are created and the LOE status is continuously monitored. It is to be noted that the figure is plotted on a scale starting with only the investigation time for alerts with no waiting time in the queue (for situation with no backlog of alerts). Alert investigation for the *identification* of innocuous and significant alerts is a relatively fast decision-making process. Hence, there are no alerts at the end of each hour, whose analysis has started but not completed. As a result, an alert is considered to be either analyzed or it remains backlogged in the queue at the end of each hour. It should be understood that once a significant alert is identified it could take several hours to find a mitigation strategy by another CSOC team, however, that process is beyond the scope of this paper.

Obtaining a real data set for a research study is a major obstacle for researchers in the field of cybersecurity due to confidentiality reasons. There are very few studies in recent cybersecurity literature where real-world data is used. Authors in [37] and [38] studied cyber-incident data from the US Department of Energy while authors in [39] studied a data set published by Privacy Rights Clearinghouse for large cyber breaches. A Poisson distribution was found to be the best fit for describing the cyber-incident arrivals in these data sets. In the experiments presented in this paper, adverse events are generated using a Poisson arrival process. The alert analysis process at each CSOC location is simulated using the inputs and uncertainty in the form of adverse events as shown in Figure 3. The algorithm for the alert analysis process is given in [2]. Performance metrics (avgTTA/hr) from all CSOC locations are obtained as the outputs of the CSOC system. The backlog of alerts pertaining to the avgTTA/hr value from each of the CSOC locations is then provided to the optimization model given below. It is to be noted that for a real-world implementation, the simulation model can be replaced by the actual alert analysis processes at the CSOC locations and the avgTTA/hr values would be provided to the RL agent/optimization model.

3.3 Optimization Model

The optimization model obtains the number of alerts that are backlogged at each of the CSOC locations from the simulation model. The objective of this model is to provide alert workload re-allocation decisions to the simulation model as shown in Figure 3. In real-world, these decisions are provided to the CSOC managers for their approval. The backlog numbers from the respective CSOC locations are provided to the RL agent. The decision-making framework provided by the RL agent is based on the principles of stochastic dynamic programming, whose formulation for adaptive alert management is presented next.

3.3.1 Stochastic Dynamic Programming Formulation

Figure 4 shows the various elements used in the stochastic dynamic programming formulation, which are explained as follows.

State variable, also known as the state of knowledge, contains all the information that is needed to make a decision (shown as squares in Figure 4). It is a n-dimensional vector, $B_t = (X_{1,t}, \dots, X_{n,t})$, where $X_{i,t}$ is the number of alerts that are backlogged at CSOC location i at time t.

Decision variable represents how the process is controlled. It is represented as a vector, $g_t = (y_{1,2,t}, \dots, y_{n,n-1,t})$, where $y_{i,l,t}$ represents the alert workload re-allocated from location i to location l for investigation between time t and $t+1$. g_t is the action taken at time t in state B_t that moves the system to the post-decision state B_t^g. The sequence of decisions are shown in solid lines in Figure 4.

Uncertainty, represented by W_{t+1} and shown in a dashed line in Figure 4, is the exogenous information that is presented at each time index. It is represented by vector $(Z_{1,t}, \dots, Z_{n,t})$, where $Z_{i,t}$ is the number of alerts that are generated in addition to the baseline average number due to the adverse events at location i between time $t-1$ and t. Adverse events, in the context of the problem in this work, are presented in Section 4. These random events impact the number of alerts that are backlogged at the CSOC locations.

State transition function determines how the system moves from one state (B_t) to another (B_{t+1}) under decision g_t in the face of uncertainty W_{t+1}. This is expressed as $B_{t+1} = h^W(B_t, g_t, W_{t+1})$. Next, the concept of post-decision state (PDS) variable [5] is explained, which breaks the transition function into two parts.

Post-decision state (PDS) variable represents the state of the system after a decision is made and is represented by B_t^g. It is to be noted that the PDS variable represents the state of the system before the exogenous information, W_{t+1} arrives. Hence, the state transition function is broken into two parts as shown in Figure 4. The state transition function that moves the system from state B_t to state B_t^g is purely due to the effect of decision g_t, while the state transition function that moves the system from state B_t^g to state B_{t+1} is due the effect of the uncertainty, W_{t+1}.

Contribution function, $C(B_t, g_t)$, is the measurement of the well-being of the system. In this work, the contribution function represents a reward that is maximized by the stochastic dynamic programming algorithm at time t and is dependent on the state of the system B_t. It is to be noted that the algorithm does not have the exogenous information W_{t+1} available at the time of making the decision g_t. The most desirable state of the system is to have minimal backlog of alerts and minimal variance between the backlog numbers across all the locations at time $t+1$. Hence, under an adverse event when the state of the system shows a high backlog of alerts at a CSOC location, the alert workload is re-allocated to other locations such that the backlog numbers

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

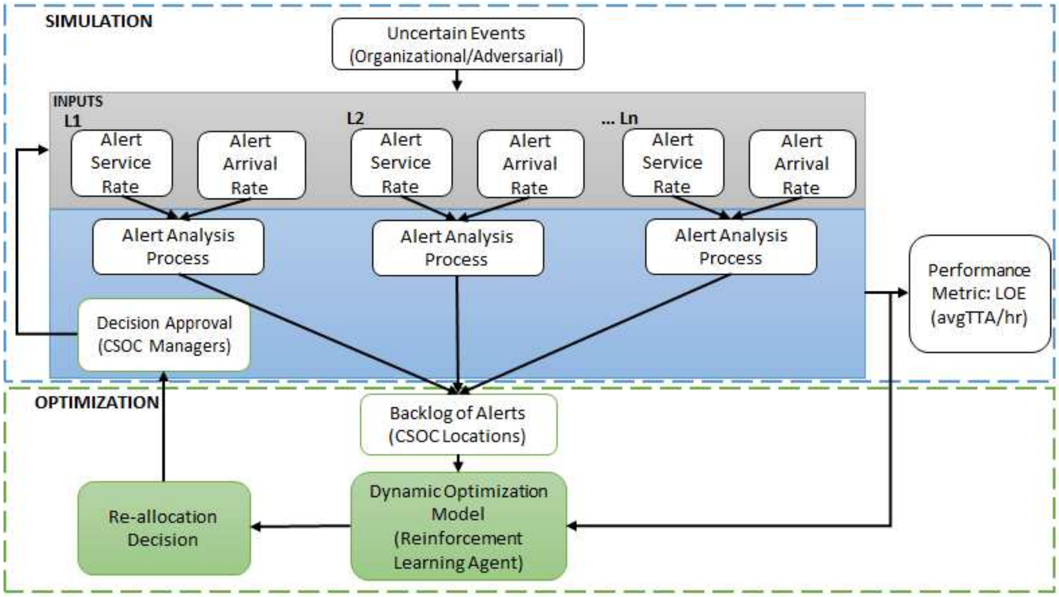


Fig. 3. Adaptive Alert Management Model Framework

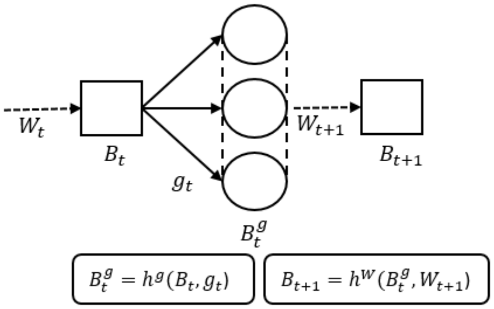


Fig. 4. Elements of the Adaptive Alert Management Formulation

are uniformly balanced among all the CSOC locations. The re-allocation decision is dependent on the current backlog numbers at the other locations. For instance, re-allocating alerts to a location that has a significant backlog of alerts at time t will result in a further increase in the backlog of alerts at the respective location at time $t+1$. The contribution function, $C(B_t, g_t)$, where decision g_t is taken in state B_t, is calculated as follows. If the backlog $X_{i,t}$ is zero for a CSOC location i then the linearly normalized value represented by $f_{i,t}$ (on a 0-1 scale) of the backlog at $X_{i,t} = 0$, is $f_{i,t} = 1$. A threshold value of backlog of alerts is calculated using the threshold value of avgTTA/hr, and value $f_{i,t}$ for $X_{i,t} \geq \textit{backlog threshold}$ is $f_{i,t} = 0$. In order to balance the LOE of the CSOC locations measured in terms of the backlog of alerts in the system state, an absolute difference among the normalized backlog numbers for all pairs of locations is taken into account in the contribution function. This helps in preventing situations of high variability among the LOE of the different CSOC locations. The contribution function is defined as follows:

$$C(B_t, g_t) = \sum_{i=1}^{n} f_{i,t} - \sum_{i=1}^{n-1} \sum_{k=i}^{n-1} (|f_{i,t} - f_{k+1,t}|). \quad (3)$$

It is to be noted that each CSOC location chooses it's own threshold value of avgTTA/hr based on the criticality of it's operations before normalizing the score, thereby, taking into consideration the importance of their respective performances.

Objective function in a stochastic dynamic programming algorithm provides the (near-) optimal policy which maximizes the well-being of the system. The objective is to maximize the rewards (Equation 3) over the time horizon of the problem. The goal is to move from one good state to another by learning the long-run total discounted values of the states $V^j(B)$ as the iteration index $j \to \infty$. This is achieved by making a decision under uncertainty to move to high valued future states at each time index t. In this work, several iterations of 14-day work cycles are simulated and using the iterative Bellman's optimality equation shown below (Equation 5 [24]), the maximized cumulative sum of discounted contributions (rewards) for the values of the system states are obtained during the learning phase of the algorithm (explained in the next section). The equations used in the algorithm are as follows:

$$V^j(B_{t-1}^g) = (1 - \alpha^j)V^j(B_{t-1}^g) + \alpha^j \eta^j \quad (4)$$

$$\eta^j = \max_{g_t \in G_t} \{C(B_t, g_t) + \beta V^j(B_t^g)\}, \quad (5)$$

Algorithm 1: Dynamic Optimization Learning Phase

```
Input: Number of iterations for learning $J = 336,000$, % of iterations
       for exploration phase $m$, discount parameter $\beta = 0.9$, initial
       learning parameter $\alpha^0 = 0.8$, and time at the end of horizon $T$.
Output: Long-run state values $V(B), \forall B$
Initialize $V(B) = 0, \forall B$
Initialize state $B_1 = (0, \ldots, 0)$ /* no backlog across all locations */
$M = m * J$ /* number of iterations in the exploration phase */
for $j = 1, 2, \ldots, J$ do
    $V^j(B) = V^{j-1}(B), \forall B$
    Decay the learning parameter, $\alpha^j = \frac{\alpha^{j-1}}{1+e}$, where
    for $t = 1, 2, \ldots, T$ do
        if $(j \leq M)$ /* Exploration Phase */, then
            Pick an arbitrary action $g_t$
            Compute $C(B_t, g_t)$ using Equation 3
            $\eta^j = C(B_t, g_t) + \beta V^j(B_t^g)$
        else if $(j > M)$ /* Exploitation Phase */, then
            $\eta^j = \max_{g_t \in G_t}\{C(B_t, g_t) + \beta V^j(B_t^g)\}$ /*
              $C(B_t, g_t)$ is computed using Equation 3 */
        end
        end
        $V^j(B_{t-1}^g) = (1 - \alpha^j)V^j(B_{t-1}^g) + \alpha^j \eta^j$ /* PDS value */
        Generate $W_{t+1}$ /* uncertainty through simulation or
          real-world */
        $B_{t+1} = h^W(B_t, g_t, W_{t+1})$ /* state transition function */
    end
    $MSE_j = \frac{\sum_{a=1}^{j}(V^a(B_{t-1}^g) - \eta^a)^2}{j}$ /* MSE */
end
return $V(B), \forall B$
```

Algorithm 2: Dynamic Optimization Learned Phase

```
Input: Value of the states $V(B)\forall B$ from the dynamic optimization
       learning phase (Algorithm 1), state $B$ from
       simulation/real-world at any given time.
Output: Action $g$, given state $B$
$\eta = \max_{g \in G}\{C(B, g) + \beta V(B^g)\}$ /* $C(B, g)$ is computed using
  Equation 3 */
Record action $g$
return g for given state $B$
```

where η^j is a sample realization of the value of the system state, G_t is the set of all feasible actions, β is the fixed discount factor that gives a weight to the future rewards and allows the state values to converge in the long run, and α^j is the learning parameter that is decayed gradually over several iterations [40]. The values of the discount and learning parameter, and the decay scheme are explained in Algorithm 1. The alpha-decay parameters are chosen such that α^j is never reduced to value 0, which will result in apparent convergence of Equation (4). The value of the PDS variable B_{t-1}^g is updated after reaching state B_t^g (Equation 4 [5]).

3.3.2 Stochastic Dynamic Programming Algorithm

The steps of the stochastic dynamic programming algorithm for making dynamic alert workload re-allocation decisions across the CSOC locations are given in Algorithm 1 and Algorithm 2. The adaptive alert management algorithm is executed in three phases, namely, **exploration**, **exploitation (learning)**, and **implementation (learned)**, which are explained below.

During **exploration** (Algorithm 1), various system states are visited in order to improve the estimates of the values of being in those respective states. This is achieved by either randomly selecting the next state after updating the value of the current state or by making a random (non-optimal) decision that moves the system to the next state. While exploring, Equation 4 (Algorithm 1) is executed to update the value of the current system state. Equation 5 (Algorithm 1) for determining the next system state is executed by replacing the max operator with a random decision on the number of alerts to re-allocate and the location(s) where the alerts are re-allocated for investigation between time t and $t + 1$. It is to be noted that the additional analyst time that is kept for non-alert analysis related activities and the manager's time at the affected location are utilized prior to receiving help from resources at other locations. The algorithm is initiated with all $V^0(B) = 0 \; \forall B$ at $j = 0$ and the values of the states are populated as they are visited. The values of $V^j(B_t^g)$ in Equation 5 and $V^j(B_{t-1}^g)$ in Equation 4 (Algorithm 1) are obtained from the previously stored values if the state was visited earlier, otherwise the value is 0. During exploration, there are various methods for determining how to collect information at various iterations of the algorithm such as pure exploration, ϵ-greedy exploration, and interval estimation [5]. Pure exploration consists of taking random actions from a state that will take millions of iterations for state values to converge. Likewise, interval estimation method will require an action to be taken many times from a state in order to estimate the value of the state with great accuracy. In this work, ϵ-greedy exploration, is used for a set number of iterations, which is based on the size of the state-space. Usually 10-20% of the iterations are used for exploration. The value of ϵ_j is decayed from 0.9 to 0 within the number of iterations allocated for exploration using the same scheme as in [40] that is used for α^j decay. Random actions are chosen with a probability $p_j < \epsilon_j$, and greedy actions with Equation 5, otherwise. The value of probability p_j is a uniformly distributed random number that is generated for every iteration j of the exploration phase using $U[0, 1]$. Once exploration is stopped, the algorithm moves into an exploitation phase, which is the greedy phase. The advantage of ϵ-greedy exploration is that it ensures a smooth transition to the exploitation phase as ϵ_j is decayed, and provides the flexibility to set the number of iterations for exploration, unlike the other methods that require a very large number of iterations.

During **exploitation** (Algorithm 1), the algorithm is executed with Equation 5 to take optimal decisions (greedy) at time t, based on the available estimates of the value of the states visited during exploration. Using η^j from Equation 5, the value of the previous post decision state is updated at time t as per Equation 4. The algorithm is executed over a large number of iterations to capture evolving patterns in uncertainty. Finally, the algorithm reaches a stage where it is capable of making optimal (or close to optimal) decisions that move the system from one good state to another in terms of uniformly balancing the individual CSOC performances (LOE) between the locations by minimizing the variance in the normalized backlog numbers across all the CSOC locations. Learning is stopped when convergence of the value of the states is achieved. This is measured by the mean-squared error (MSE) of the stochastic gradient [5] at the j iteration, which is given as follows.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

$$MSE_j = \frac{\sum_{a=1}^{j}(V^a(B^g_{t-1}) - \eta^a)^2}{j} \qquad j \neq 0 \qquad (6)$$

where the variables are as defined earlier. After learning the values of the system states ($V(B), \forall B$) at the end of the exploitation phase (Algorithm 1), an optimal decision g is chosen for a state B obtained from simulation/real-world at any given time using Equation 5 in the **implementation (learned)** phase (Algorithm 2).

4 Experimental Set-up

The following section presents the experimental set-up that is used in this work. A baseline case is established for each of the CSOC locations with a pre-determined (acceptable) avgTTA/hr. The level of operational effectiveness (LOE) for each of the CSOC locations is deemed to be ideal at their respective avgTTA/hr values. Three locations ($n = 3$) are considered for the experiments presented in this work and are represented by L1, L2, and L3 respectively. Table 1 shows the inputs for the *baseline case* for the three locations.

Inputs used in the experiments were obtained from the literature and numerous conversations with the CSOC operators. For example, one of the co-authors in this paper is from the Army Research Lab (ARL) and has a liaison with ARL's CSOC. The inputs for one of the locations, L1, are used from Shah et al. [3], while the inputs for the other two locations are chosen such that the alert arrival rate, service rate, baseline avgTTA/hr, and threshold avgTTA/hr values vary from one another. The number of sensor clusters used for alert generation at L1, L2, and L3 locations are 10, 15, and 15, respectively. Each cluster has, on average, 10 sensors. Sensors are typically clustered based on types, and are formed such that the expected number of alerts generated (from historical data) are uniformly balanced among them. This research uses an exponential distribution with time between alert arrivals of 18.8 seconds/sensor cluster (Poisson arrival rate) as in [3]. It is to be noted that the analysts are staffed in each shift such that the alert service rate (μ) is slightly higher than the alert arrival rate (λ), i.e. $\rho = (\lambda/\mu) < 1$. A larger difference between λ and μ would mean that the analysts are idling, which is not customary at a CSOC. Hence, there are 10 analysts (with 80% of their time spent on alert investigation and the remaining 20% on training, report writing, and updating alert signatures) working on 10 sensor clusters at location L1 such that $\rho < 1$. Similarly, there are 15 analysts working on 15 sensor clusters at locations L2 and L3 with their respective ρ values < 1. The baseline avgTTA/hr value of 1 and a threshold avgTTA/hr value of 4 are chosen for location L1. Different values are chosen for the acceptable LOE (avgTTA/hr) at locations L2 and L3, which indicates the criticality or importance of the operations of one CSOC location compared to another. The baseline avgTTA/hr value of 0.5 (1) and a threshold avgTTA/hr value of 2 (3) are chosen for location L2 (L3). The above values for setting up the baseline cases were obtained through discussions with various CSOCs and through literature survey [3]. Based on the above inputs, the alert analysis process is simulated to reach steady state conditions using the algorithm in [2]. The average alert queue lengths based on the baseline avgTTA/hr values for locations L1, L2, and L3 are established at 1175 alerts, 1440 alerts, and 3000 alerts respectively. Similarly, the average alert queue lengths based on the threshold avgTTA/hr values for locations L1, L2, and L3 are established at 4350 alerts, 5760 alerts, and 8470 alerts, respectively. The above values are determined so that deviations from the baseline numbers, as the time progresses, can be displayed using the color-coded representation of the LOE status (Figure 2 [2]) and corrective actions are initiated by the CSOC locations. Next, a normalized value for the backlog of alerts for each CSOC location is determined. For location L1, a backlog $>$ 3175 (threshold - baseline = 4350-1175) alerts is given a value $f_{i=L1,t} = 0$ and a backlog $<$ 1175 alerts is a given a value $f_{i=L1,t} = 1$ at time index t. Similarly, values for $f_{i=L2,t} = 0$ and $f_{i=L3,t} = 0$ are determined at backlog $>$ 4320 alerts and $>$5470 alerts respectively. $f_{i,t}$ for both locations are given a value of 1 at or below their respective baseline alert backlog numbers. The values between threshold and baseline backlog of alerts for each of the CSOC locations are then linearly normalized. During the experiments performed, a CSOC location under an adverse event first utilizes the remaining 20% of effort available in the shift for the available analysts along with the manager's time to assist with the alert investigation. For example, this could include cancelling a training session for an analyst or postponing report writing. The above is considered to be an internal action taken by the respective CSOC location before the centralized agent would re-allocate the alert workload to the other locations, and any such additional effort used by a location is reflected as a reduction in the number of alerts in the backlog for that location.

Uncertainty occurs with a wide range of variability in the number of stochastic events, which is modeled using a Poisson distribution with a mean of seven events across the three locations in a 14-day (336 hours) period. Table 2 presents a sample of adverse events that are simulated at random during the experiments, which are obtained from [3]. The events impact the alert generation rates, and the alert investigation (service) rates. Some of the events include (1) an increase in both intensity and duration of surges in alert generation due to an attack or the restoration of a broken communication link between the sensor/IDS and the CSOC, and (2) an increase in alert investigation time due to the discovery of a new alert pattern.

In this research, the RL approach, which is based on the principles of stochastic dynamic programming is compared with rule-based and load balancing approaches. The three approaches are described below.

4.1 Reinforcement Learning (RL) Approach

The RL model is run in three stages - exploration, exploitation (learning), and implementation on sample realizations (learned). The stochastic dynamic programming algorithm learns over a large number of iterations (1,000 iterations of 14-day work cycles) during the exploration and exploitation phases (Step 1 in Algorithm 1). Figures 5(a) and 5(b) indicate the rate of α decay (learning parameter in Equation 4) and the mean-squared error (MSE) of the stochastic gradient respectively. The algorithm is tested on several types

TABLE 1
Inputs for *baseline case*

	L1	L2	L3
Number of clusters of sensors	10	15	15
Average time between alert generation (s)	Expo(18.8)	Expo(18.8)	Expo(18.8)
Number of analysts	10	15	15
% effort of analysts towards alert analysis	80%	80%	80%
Average time taken to investigate an alert T (s)	15	15	15
Baseline avgTTA/hr (hr)	1	0.5	1
Threshold avgTTA/hr (hr)	4	2	3

of uncertain disruptive events during the implementation phase (Algorithm 2) using 50 simulation runs of the 14-day work cycles to obtain 95% confidence intervals for the LOE performance metrics for the various locations. Also, for the same disruptive events in a 14-day period, a rule-based approach and two types of load balancing approaches for decision-making is implemented as explained below. In each instance of the simulation run, the alert analysis process is simulated using the algorithm in [2] over a 14-day period. Alert workload re-allocation decisions and their impact on the LOE of all the CSOC locations are recorded using the RL, rule-based, and two types of load balancing approaches.

4.2 Rule-based Approach

The authors in [41] show a comparison between the rule-based and classic machine learning systems. Rule-based systems are hand-designed programs where the system is dependent on a set of rules while in the case of the latter, the system learns from historical data. Despite the growing interest in cybersecurity research, real data sets are challenging to obtain due to the sensitive nature of the data. Hence, through conversations with several CSOC managers we obtained the real-world rules, which CSOCs typically follow to improve their performances in the presence of adverse events. The conditions for the rule-based approach are explained as follows.

1) If the LOE of a CSOC crosses into the yellow zone (Figure 2), re-allocate alert workload to other CSOC locations that are in the green zones in order to bring the LOE back into the green zone at the affected location.
2) If the LOE of a CSOC crosses into the orange zone (Figure 2), re-allocate alert workload to other CSOC locations that are either in the green or yellow zones in order to bring the LOE back at least into the yellow zone at the affected location.
3) If the LOE of a CSOC crosses into the red zone (Figure 2), re-allocate alert workload to other CSOC locations that are not in the red zones in order to bring the LOE back at least into the orange zone at the affected location.

4.3 Load Balancing Approach

In this approach, the CSOC locations are continuously monitored and alert workload re-allocation decisions are made between the locations with the worst LOE and the best LOE. The timing of the reallocation decision is an important factor to consider. For example, small differences in queue length or LOE among the locations should not trigger a re-allocation of alerts. Besides, too many re-allocation decisions are not practical to implement. Hence, two practically implementable load balancing approaches are studied below.

In Type 1 load balancing approach, as soon as either the alert arrival rate (λ) or the alert service rate (μ) is affected (i.e., the value of (ρ) goes over 1), the alert workload queue length is balanced between the locations with the best and worst LOE at the time of observation. This situation can occur at any time during continuous monitoring. Accordingly, a certain number of alerts are re-allocated from the location with the worst LOE to the one with the best LOE. It is to be noted that in this method, the baseline avgTTA and threshold avgTTA values are not taken into consideration at either of the locations and the re-allocation decision is purely based only on the number of alerts that remain in the queue (backlog number) at both the locations at the time of observation and decision making. When a decision is made, the alert queue lengths are equalized at both locations. The Type 1 load balancing approach is therefore regarded as a simple load balancing approach.

In Type 2 load balancing approach, which is similar but more intelligent than the Type 1 approach, alert workload from a CSOC location with the worst LOE is allocated to a CSOC location with the best LOE, however, the re-allocations are made such that the individual LOE values are balanced among the two locations. The LOE of all the CSOC locations are continuously monitored and at the top of each hour, a decision to re-allocate alerts is made for the CSOC with the worst LOE to the one with the best LOE. The number of alerts to re-allocate, unlike Type 1 approach, may not necessarily result in an equalized queue length at both locations, and is determined such that both locations have an uniformly balanced LOE. Another important difference between Type 1 and Type 2 load balancing approaches is that in Type 2 load balancing approach the threshold and baseline avgTTA values are taken into consideration at both locations before making a decision on the number of alerts to be reallocated. It is to be noted that both approaches make a static reactive decision at the time of observation and do not take future uncertainties into consideration.

Figure 6 shows the temporal patterns of a sample realization used in one of the experiments presented in this work. The sample realization consists of seven random adverse events that were generated in the 14-day period. Figure 6 shows the time, location, and average number of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

TABLE 2
Adverse events for *experiments*

Event 1 (E1)	30% increase in alert generation for 8 hours
Event 2 (E2)	40% increase in alert generation for 8 hours
Event 3 (E3)	New vulnerability that increases alert investigation time by 5 times for 12 hours
Event 4 (E4)	30% increase in alert generation for 12 hours
Event 5 (E5)	Communication breakdown between sensors/IDS and CSOC for 12 hours

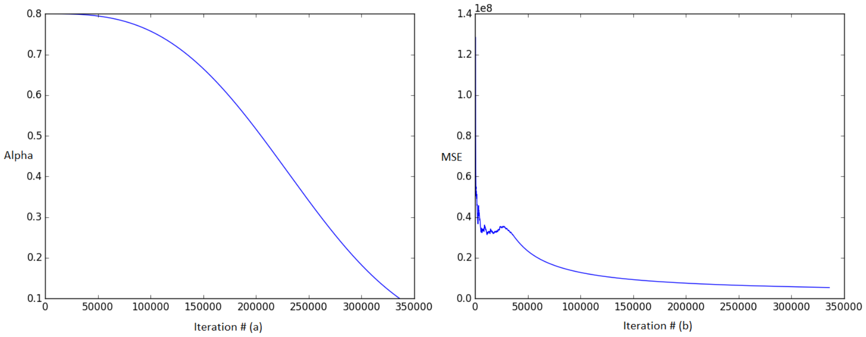


Fig. 5. (a) Learning parameter decay, and (b) Mean-squared error of the stochastic gradient

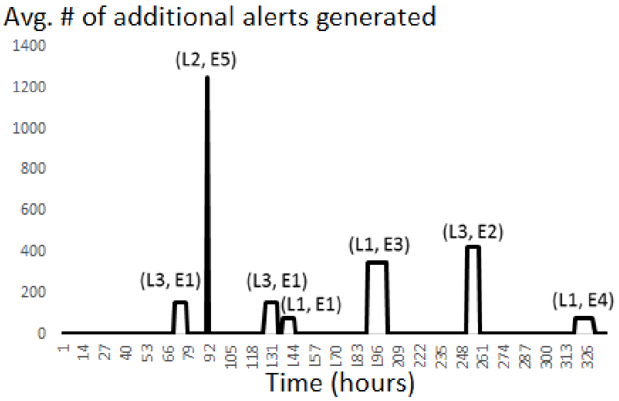


Fig. 6. Temporal Patterns in Adverse Events

additional alerts generated for each event. For example, the first event occurs in L3 (location 3) and the event type is E1 (see Table 2). All three approaches - RL, rule-based, and the two types of load balancing, are tested using the above sample realization of uncertain adverse events. The results are analyzed and comparisons are presented in the next section.

5 Analysis of Results

The following section presents the results from the experiment conducted with the sample realization (Figure 6) described in the previous section. Figure 7 shows the LOE of all the 3 locations without re-allocation of alerts as a basecase to compare the three approaches mentioned above. It is to be noted that whenever an adverse event occurs, a CSOC will first utilize the additional 20% effort in the form of additional analyst time by canceling/delaying the non-alert analysis related tasks and by calling in a manager to assist with the alert analysis task. After completely depleting its own resources, the affected location follows the decisions by the centralized agent, which re-allocates the remaining alert workload to the other locations. Figure 8 (a-c), Figure 9 (a-c), Figure 10 (a-c), and Figure 11 (a-c) show the resources that were utilized at the respective locations to investigate the additional alert workload distributed to them by either of the remaining locations in the rule-based, Type 1 load balancing, Type 2 load balancing, and RL approach, respectively. Figure 8 (d-f), Figure 10 (d-f), and Figure 11 (d-f) show the avgTTA/hr plot against the background of the color-coded LOE representation for each of the locations in the rule-based, Type 1 load balancing, Type 2 load balancing, and RL approach, respectively.

In the rule-based approach, the avgTTA/hr of all the three locations (L1, L2, and L3) climb up to the top of the green zone (as shown in Figure 8 (d-f)) with the arrival of the first four events. Later, the avgTTA/hr of L1 crosses into the yellow zone when faced with the fifth event and as a result the alert workload is re-allocated to L2 to bring the avgTTA/hr of L1 back into the green zone, as shown by the spike in resources allocated by L2 in Figure 8 (b). Next, avgTTA/hr of L2 crosses into the yellow zone. L1 and L3 allocate their resources to investigate the alert workload from L2, as shown in Figure 8 (a) and Figure 8 (c), respectively, in order to bring the avgTTA/hr of L2 back into the green zone. As a result, avgTTA/hr of L3 crosses into the yellow zone and is assisted by re-allocating the alert workload to L1 in order to bring it back into the green zone. However, this pushes the avgTTA/hr of L1 into the yellow zone and correspondingly resources from L2 are utilized. Later, L2 finds its avgTTA/hr in the red zone while trying to assist L3 and by allocating its resources, L1 crosses into the red zone, which initiates the alert workload re-allocation to L3. Hence, it can be observed that there is a tug-of-war among locations for resources based on the set of rules using this approach. Finally, with the seventh adverse event (faced by L1), the avgTTA/hr of L1 crosses into the red zone, and by allocating L2 and L3 resources to assist L1, both L2 and L3

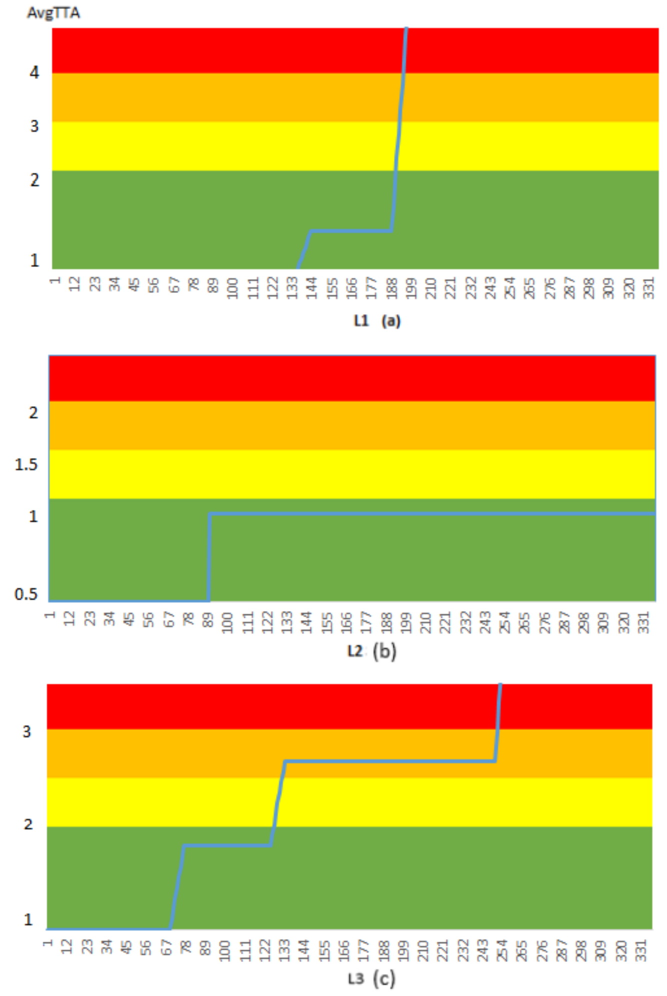


Fig. 7. Basecase (without re-allocation of alerts): (a-c) avgTTA (LOE) of 3 locations

find their respective avgTTA/hr climb up to the red zone towards the end of the 14-day period.

For the Type 1 load balancing approach, Figure 9 (d-f) shows the dynamic behavior of avgTTA metric for all the three (L1, L2, and L3) CSOC locations for the 14-day period. The arrival rate or the service rate of alerts is impacted when a CSOC location is faced with an adverse event. As a result, when the traffic intensity is higher than 1 ($\rho = \lambda/\mu 1$), alert workload re-allocation decisions are made in which the alert workload is equally distributed between the locations with the best and worst LOE. This method only looks at the backlog length (i.e., the number of alerts in the queue) at both (best and worst) CSOC locations and re-allocates alert workload from one location to another such that the backlog length is equal at both locations at the time of decision-making. It is to be noted that this method is myopic and does not take into account future adverse events that may occur at any of the CSOC locations. For instance, with the onset of event E1 at location L3, a part of the alert workload was re-allocated to location L1. However, soon after receiving the additional workload from location L3, L1 also faces an adverse event (E1). As a result, avgTTA/hr crosses into the yellow zone at location L1 (Figure 9 (d)).

It can also be observed from Figure 9 (a-c) that a large number of alerts are re-allocated among the locations every time a re-allocation decision is made. This is due to the fact that decision-making does not take into consideration the avgTTA baseline and threshold values at the CSOC locations; instead it only looks at the alert backlog at the CSOC locations. As a result, the CSOC locations have a higher number of alerts in the queue, which increases the avgTTA/hr value for all the locations. It is also observed that under this approach, the LOE at all the CSOC locations crosses into the red zone in the 14-day period (Figure 9 (d-f)).

The performance of Type 2 load balancing approach, which allocates alerts for investigation from a CSOC location with the worst LOE to the one with the best LOE by taking into consideration the values for baseline avgTTA and the threshold avgTTA at both locations such that the LOE for both locations are balanced, is shown in Figure 10. Since the LOE at L2 was the best among the 3 locations at the onset of the first event (E1) at location L3, additional alerts are first allocated from L3 to L2 for investigation. When event E5 occurs, which generates a large number of alerts at location L2, additional alerts are allocated from L2 to L1 for investigation. It is to be noted that since this method is static (and hence, myopic), it does not consider future uncertainties. As a result, it allocates a larger number of alerts from locations experiencing an adverse event to the locations with better LOE, compared to the RL approach (explained next). For example, when events 4 and 5 occur (see Figure 6) at location L1 at a time when the backlog of alerts was already high due to the additional alerts allocated from location L2 in the prior event, the LOE deteriorates at location L1 and crosses into the orange zone. With a larger difference among the LOE of locations L1 and L3 and with the final event (E4) occurring at location L1, a large numbers of alerts are allocated from L1 to location L3, which in turn pushes the avgTTA, into the red zone (see Figure 10 (d-f)).

In summary, it is observed from both types of load balancing approaches that (1) since these methods are myopic, i.e. they do not take future uncertainties into consideration, larger numbers of alerts are allocated from locations which have the worst LOE to the ones with the best LOE in order to uniformly balance the alert workload or the LOE among them (see Figure 9 (a-c) and Figure 10 (a-c)), and (2) the avgTTA/hr crosses into the red zones at all of the locations using Type 1 load balancing approach, and at locations L1 and L3 using the Type 2 load balancing approach. Although Type 2 load balancing is overall better than Type 1 approach, it was still not better than the RL approach given below.

In the RL approach shown in Figure 11 (d-f), the avgTTA/hr of all the three locations follow a similar trend and is uniformly balanced throughout the 14-day period. In particular, LOE is maintained at similar levels in the green zone for all the three locations within the same time-period (from the start of the 14-day period until the occurrence of the fifth event). Similar observations are made for the LOE in the yellow and orange zones. In order to maintain the operational effectiveness of all the locations at similar (near-equal) levels, the number of times alert workload re-allocation decisions are made among the locations (as shown in Figure 11 (a-c)) using the RL approach is higher when compared with the rule-based approach. However, it

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

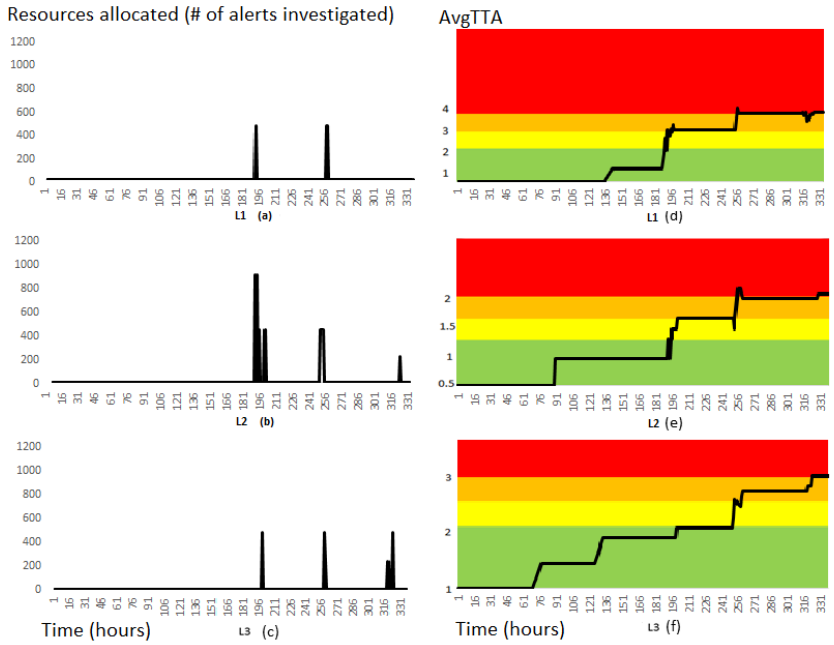


Fig. 8. Rule-based approach: (a-c) additional alerts investigated, (d-f) avgTTA (LOE)

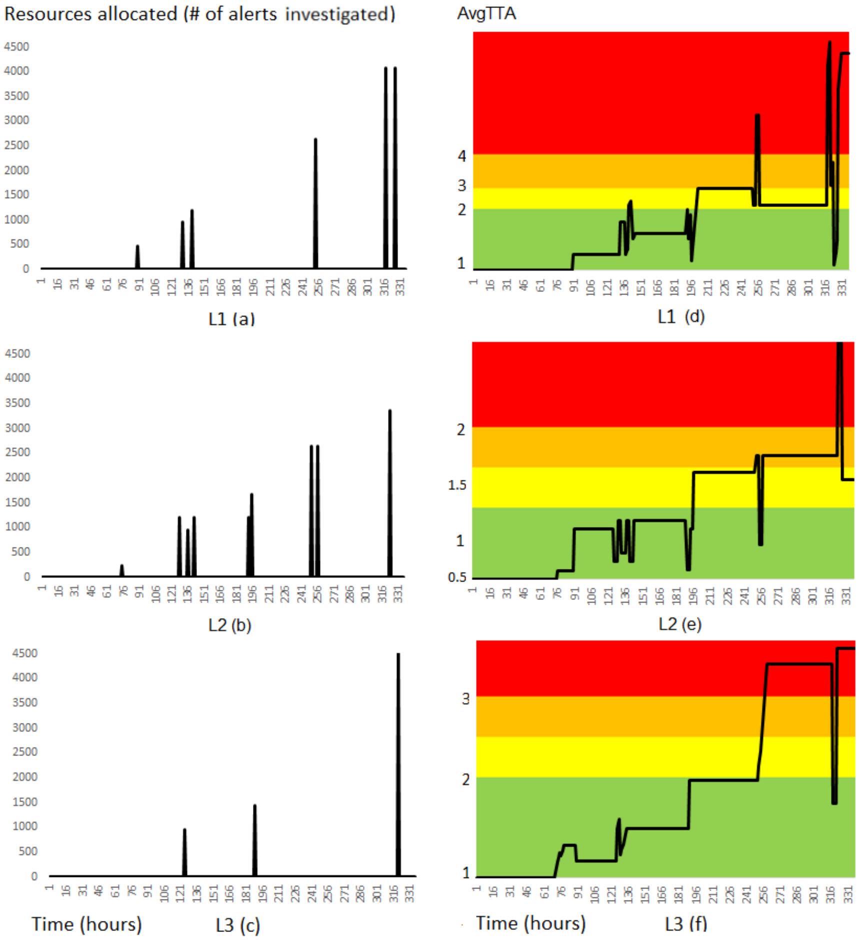


Fig. 9. Type 1 Load Balancing approach: (a-c) additional alerts investigated, (d-f) avgTTA (LOE)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

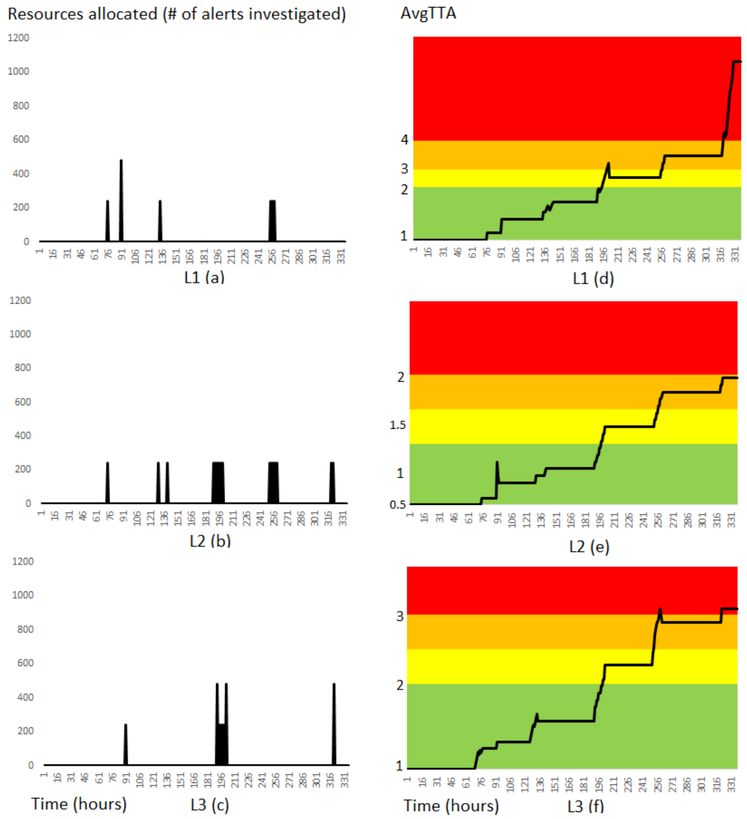


Fig. 10. Type 2 Load Balancing approach: (a-c) additional alerts investigated, (d-f) avgTTA (LOE)

is to be noted that for most of the re-allocation decisions, the number of alerts that are re-allocated in the RL-approach is smaller when compared with the rule-based approach. The reason for the above is that through RL, the system has learned to make corrections that would keep the CSOC in good states (where individual avgTTA/hr (LOE) are balanced among the locations) in the long run. In summary, the RL-based re-allocation decisions are made more often and in smaller proportions compared to a reactive rule-based approach that waits for the LOE to deteriorate before making a re-allocation decision. Figure 12 (a-d) shows a comparison of the amount (%) of time that the entire CSOC system (all locations) spent in a centralized mode when resources were allocated by either of the CSOC locations to investigate additional alerts from the other location(s), and in a distributed mode under all the four approaches for the 14-day period.

In this experiment, with adverse events generating a greater number of alerts than the total capacity to analyze them over a 14-day period, the avgTTA/hr of each location is shown to approach the threshold (red zone) towards the end of the 14-day period (see Figure 11 (d-f)). However, the sudden degradation of the LOE as witnessed in the rule-based approach is avoided using the RL approach, for example, when the avgTTA/hr for L1 and L3 deteriorated from the green zone to the orange zone within a day. Several other experiments (with sample realizations) were conducted for a 14-day period and the observations from the results were similar. In summary, the RL approach outperformed both rule-based approach and static load balancing approaches, by maintaining a lower value for avgTTA/hr, and by uniformly balancing the individual avgTTA/hr performances (LOE) between the CSOC locations by taking (near-) optimal decisions on the timing, number of alerts to re-allocate, and the selection of the locations to which the alerts are re-allocated.

Figure 13 shows the % of time avgTTA/hr spent in each color-coded zone for all the 3 locations under the baseline case and the above approaches. The results clearly indicate that in the RL approach shown in Figure 13 (m-o), the % of time the CSOC spent in a certain color-coded LOE zone is balanced among the 3 locations. Figure 13 (b) for L2 is in the green zone for the basecase, however, additional alerts from L1 and L3 were allocated to it. Hence, the % of green zone is reduced in Figure 13 (e) and Figure 13 (h). Comparing Figure 13 (d-f), Figure 13 (g-i), and Figure 13 (j-l) indicates that both rule-based approach and static load balancing approach fail to minimize the total backlog and balance the backlog across the locations, whereas the RL-approach is both dynamic and adaptive in achieving the objective of the research.

5.1 Scalability Analysis

The number of system states defines the computational complexity of the dynamic programming algorithm. The system state vector used in the experiments is 3-dimensional which represents the LOE status (measured in terms of the backlog of alerts) of the three locations. The computational complexity is $O(\nu^3)$, where ν is the maximum number of discretizations for the backlog of alerts. For the backlog

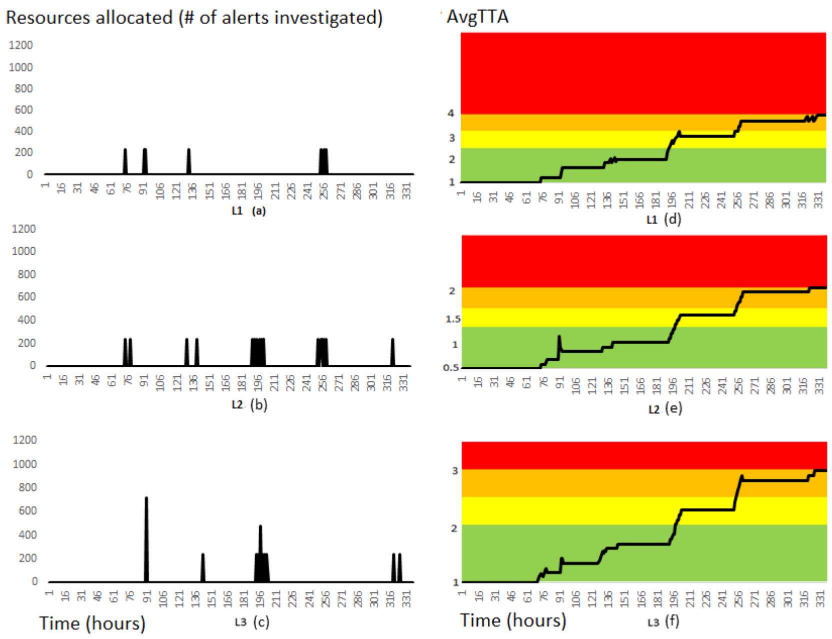


Fig. 11. RL approach: (a-c) additional alerts investigated, (d-f) avgTTA (LOE)

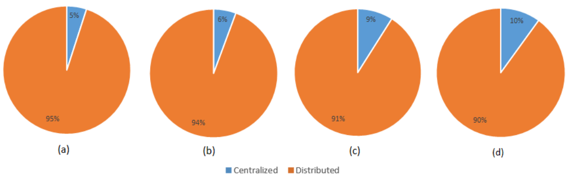


Fig. 12. Comparison of time spent in centralized and distributed modes in a 14-day period: (a) Rule-based approach, (b) Type 1 load balancing approach, (c) Type 2 load balancing approach, and (d) RL approach

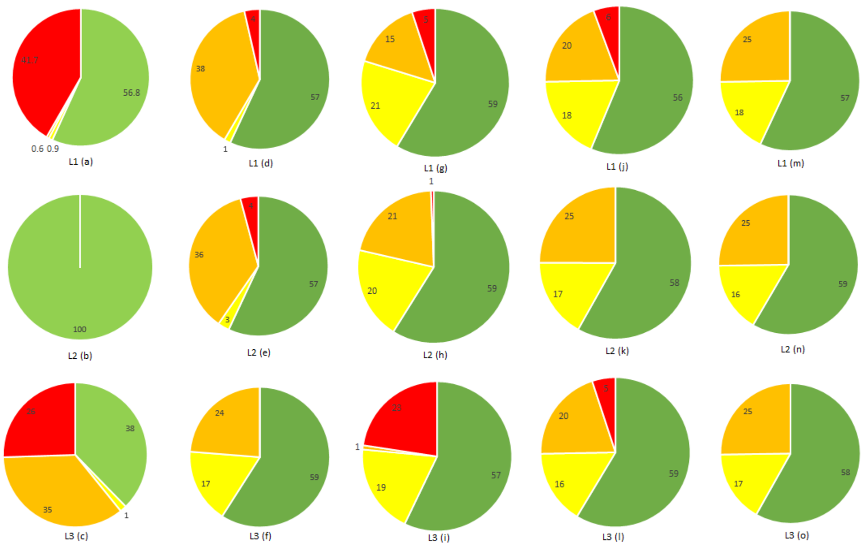


Fig. 13. % of time LOE in the respective zones over a 14-day period: (a-c) Basecase, (d-f) Rule-based approach, (g-i) Type 1 load balancing approach, (j-l) Type 1 load balancing approach, and (m-o) RL approach

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

parameter $X_{i,t}$ for location 1, an upper bound at 5000 was chosen with a step size of 50 representing 100 values. Similarly, for the backlog parameter y_t (Z_t) for location 2 (location 3), an upper bound at 6000 (9000) was chosen with step size of 50 representing 120 (180) values. As a result, the state space consisted of more than two million states (100 X 120 X 180). As the number of CSOC locations increase, the dimension of the system state vector will increase. Hence, a new representation of the system state must be used to avoid the curse of dimensionality for the state space [5]. In what follows, we present i) a modified state variable, ii) results from the experiment with the three locations conducted using the new state variable, and iii) a scalability study with nine locations.

In order to avoid the state space explosion, which may occur for organizations with more than 4 or 5 locations, a new system state representation is modeled and tested. The normalized backlog numbers from all the CSOC locations are aggregated and a mean value is calculated at the top of each hour. Next, the maximum deviation from this mean value by each normalized backlog number is calculated for all the locations. This information is then presented as the state of knowledge to the RL agent to learn the (near-) optimal actions that not only minimizes the mean of the normalized backlog numbers but also the variance among them. The normalized backlog values between [0,1] are used because each location has different alert arrival rates and threshold for LOE (1 indicates backlog at or below the baseline number, and 0 indicates that backlog is at the threshold value or higher).

State variable is defined as a 2-dimensional vector, $B_t = (M_t, D_t)$, where

$$M_t = \frac{\sum_{i=1}^{n} f_{i,t}}{n} \tag{7}$$

and

$$D_t = maximum(abs(M_t - f_{1,t}), \ldots, abs(M_t - f_{n,t})) \tag{8}$$

Algorithm 1 is executed with the new state variable as explained above and state values are obtained. It is to be noted that with a 2-dimensional state vector and with 3 decimal places for the mean and the maximum deviation numbers, the learning algorithm converged faster than the previous experiment with individual backlog numbers as state variables. Figure 14 (a) shows the results obtained from the run of Algorithm 2 using the same random seed as used in the earlier experiment with the three locations. The figure shows the normalized avgTTA across all the locations with the color-coded status. The figure plots the average, best, and worst avgTTA observed among the three locations. Figure 14 (b) shows the corresponding values of the above metrics from the previous experiment conducted with the original state representation (with the actual backlog numbers for all CSOC locations). It is to be noted that the results obtained with aggregated state (mean of normalized backlog values from all locations) are statistically insignificant from the results obtained using individual backlog numbers as state variables. Thus, the new aggregated representation of the system state is able to guide the RL agent to make good decisions. It must be understood that individual backlog numbers are still maintained for each location, which is used in calculating the aggregated state-space. The actions, however, are not aggregated and are taken at the individual locations because only a small set of feasible actions are permissible for a given state. The impact of an action on the backlog is first measured at the individual locations. Later, aggregated mean of the normalized backlog numbers from all locations is reported as the system state. Hence, there is no loss of fidelity due to the aggregated state. Thus, the method of state aggregation allows for higher number of locations to be tested without any state-space explosion issues.

To demonstrate scalability of the new state aggregation framework with the above new 2-D state vector, nine locations are simulated. The three threshold backlog numbers from the experimental set-up in Section 4 are assigned to the nine CSOC locations (one threshold backlog number to exactly three locations). Results from one instance of the adverse events across the nine locations is shown in Figure 15. It is to be noted that the best observed avgTTA and the worst observed avgTTA among all the locations remain very close to the average observed throughout the time-horizon of 14 days. The RL agent with the new state variable is able to re-allocate alert workload and balance the LOE performance among all of the nine distributed CSOCs. The above improvement has allowed for significant increase in the practicality aspect of this research by adding multiple locations of the CSOC whose LOE can be balanced.

6 Conclusions and Future Work

The paper presented a novel RL-based dynamic and adaptive decision-making framework for a centralized agent to autonomously re-allocate workload in order to balance the LOE performance among several distributed CSOCs of an organization. Results of the new framework are compared to baseline (no-reallocation), a rule-based reallocation strategy, and two types of load balancing strategies, and it is observed that the RL-based approach is superior in making the non-trivial decisions of when, how much, and whom to reallocate the alerts in order to minimize the total backlog of all CSOCs and to uniformly balance the LOE performance among all the CSOCs. Several uncertain events were simulated along with several 14-day simulation runs of alert generation in order to establish the superior performance of the RL-approach over other approaches. The paper serves as a paradigm shift in autonomous decision-making using intelligence for the purpose of uniformly balancing the LOE performance among all the distributed CSOCs.

As a part of future work, it would be interesting to combine the RL framework developed in this research for utilizing the resources available at various CSOC locations with the RL framework for managing the on-call analyst resources [3] to maintain an optimal LOE at each CSOC location. By considering hourly batches of alerts, which are collected/triaged from the sensors, a RL model could be tested as a part of the future work to re-distribute the number of batches that are backlogged at the CSOC locations. Multiple-period look-ahead tree search is another extension of the solution strategy presented in this paper. However,

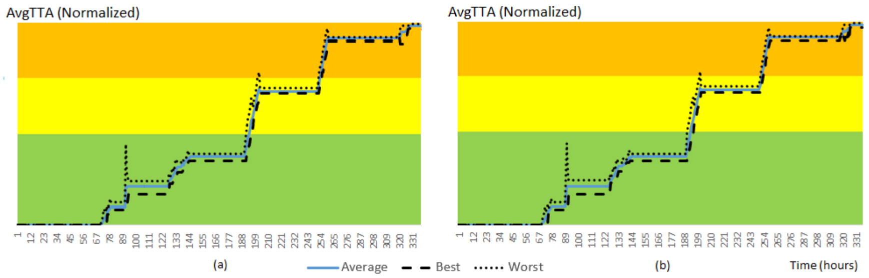


Fig. 14. Comparison of results with 3 locations using different state representations

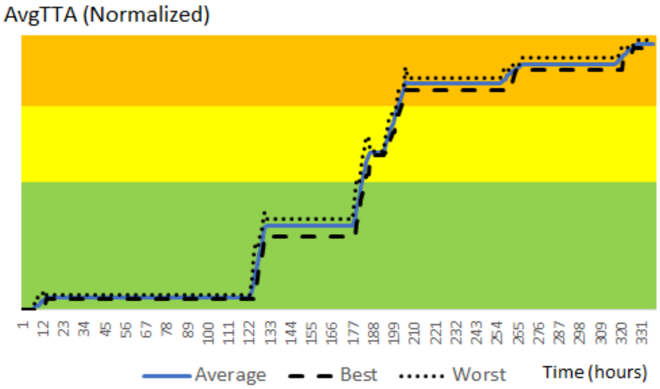


Fig. 15. Results from scalability study with 9 Locations

such a tree search would be computationally very expensive and sparse sampling techniques may have to be employed.

Acknowledgments

The authors would like to thank Dr. Cliff Wang of the Army Research Office for the many discussions which served as the inspiration for this research. The authors would also like to thank Dr. Sabrina De Capitani di Vimercati for her comments that improved the clarity of the article.

References

[1] R. Ganesan, S. Jajodia, and H. Cam, “Optimal scheduling of cybersecurity analyst for minimizing risk,” *ACM Transactions on Intelligent Systems and Technology*, vol. 8, no. 4, Feb. 2017.

[2] A. Shah, R. Ganesan, S. Jajodia, and H. Cam, “A methodology to measure and monitor level of operational effectiveness of a CSOC,” *International Journal of Information Security, Springer*, vol. 17, no. 2, pp. 121–134, 2018.

[3] ——, “Dynamic optimization of the level of operational effectiveness of a CSOC under adverse conditions,” *ACM Transactions on Intelligent Systems and Technology*, vol. 9, no. 5, pp. 51:1–51:20, Apr. 2018. [Online]. Available: http://doi.acm.org/10.1145/3173457

[4] R. Sutton and A. G. Barto, in *Reinforcement Learning*. The MIT Press, Cambridge, MA, 1998.

[5] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley-Interscience, 2007.

[6] M. J. West-Brown, D. Stikvoort, K.-P. Kossakowski, G. Killcrece, and R. Ruefle, “Handbook for computer security incident response teams (CSIRTs),” DTIC Document CMU/SEI-2003-HB-002, 2003.

[7] G. Killcrece, K.-P. Kossakowski, R. Ruefle, and M. Zajicek, “State of the practice of computer security incident response teams (csirts),” Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU/SEI-2003-TR-001, 2003.

[8] S. Bhatt, P. K. Manadhata, and L. Zomlot, “The operational role of security information and event management systems,” *IEEE Security & Privacy*, vol. 12, no. 5, pp. 35–41, 2014.

[9] R. Bejtlich, *The tao of network security monitoring:Beyond intrusion detection*. Pearson Education Inc., 2005.

[10] T. Crothers, *Implementing intrusion detection systems*. Wiley Publishing Inc., 2002.

[11] A. Rasoulifard, A. G. Bafghi, and M. Kahani, “Incremental hybrid intrusion detection using ensemble of weak classifiers,” in *Advances in Computer Science and Engineering*. Springer, 2008, pp. 577–584.

[12] S. Northcutt and J. Novak, *Network intrusion detection, 3rd Edition*. Thousand Oaks, CA: New Riders Publishing, 2002.

[13] R. Di Pietro and L. V. Mancini, Eds., *Intrusion detection systems*, ser. Advances in Information Security. Springer, 2008, vol. 38.

[14] D. Barbará and S. Jajodia, Eds., *Application of data mining in computer security*, ser. Advances in Information Security. Springer, 2002, vol. 6.

[15] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in *Proceedings of IEEE Symposium on Security and Privacy*, May 2010, pp. 305–316.

[16] C. Zimmerman, *The strategies of a world-class cybersecurity operations center*. McLean, VA: The MITRE Corporation, 2014.

[17] A. D’Amico and K. Whitley, “The real work of computer network defense analysts: The analysis roles and processes that transform network data into security situation awareness,” in *Proceedings of the Workshop on Visualization for Computer Security*, 2008, pp. 19–37.

[18] S. C. Sundaramurthy, A. G. Bardas, J. Case, X. Ou, M. Wesch, J. McHugh, and S. R. Rajagopalan, “A human capital model for mitigating security analyst burnout,” in *Eleventh Symposium on Usable Privacy and Security (SOUPS 2015)*. USENIX Association, 2015, pp. 347–359.

[19] S. C. Sundaramurthy, J. McHugh, X. Ou, M. Wesch, A. G. Bardas, and S. R. Rajagopalan, “Turning contradictions into innovations or: How we learned to stop whining and improve security operations,” in *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*. USENIX Association, 2016, pp. 237–250.

[20] R. Ganesan, S. Jajodia, A. Shah, and H. Cam, “Dynamic scheduling of cybersecurity analysts for minimizing risk using reinforcement learning,” *ACM Transactions on Intelligent Systems*

and Technology, vol. 8, no. 1, pp. 1–21, Jul. 2016. [Online]. Available: http://doi.acm.org/10.1145/2882969

[21] D. S. Altner, A. C. Rojas, and L. D. Servi, "A two-stage stochastic program for multi-shift, multi-analyst, workforce optimization with multiple on-call options," *Journal of Scheduling*, Dec 2017. [Online]. Available: https://doi.org/10.1007/s10951-017-0554-9

[22] E. A. Newcomb, R. J. Hammell, and S. Hutchinson, "Effective prioritization of network intrusion alerts to enhance situational awareness," in *Intelligence and Security Informatics (ISI), 2016 IEEE Conference on*. IEEE, 2016, pp. 73–78.

[23] G. P. Tadda and J. S. Salerno, *Cyber Situational Awareness: Issues and Research*. Springer US, 2010, ch. Overview of Cyber Situation Awareness.

[24] R. Bellman, *Dynamic Programming*. Princeton University Press, Princeton NJ, 1957.

[25] M. L. Puterman, *Markov Decision Processes*. Wiley Interscience, New York, 1994.

[26] Y. O. Yazir, C. Matthews, R. Farahbod, S. Neville, A. Guitouni, S. Ganti, and Y. Coady, "Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis," in *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, ser. CLOUD '10, 2010, pp. 91–98.

[27] M. Duggan, J. Duggan, E. Howley, and E. Barrett, "A reinforcement learning approach for the scheduling of live migration from under utilised hosts," *Memetic Computing*, vol. 9, no. 4, pp. 283–293, Dec 2017.

[28] J. Wu, X. Xu, P. Zhang, and C. Liu, "A novel multi-agent reinforcement learning approach for job scheduling in grid computing," *Future Generation Computer Systems*, vol. 27, no. 5, pp. 430 – 439, 2011.

[29] F. Farahnakian, P. Liljeberg, and J. Plosila, "Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning," in *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*. IEEE, 2014, pp. 500–507.

[30] R. Shaw, E. Howley, and E. Barrett, "An advanced reinforcement learning approach for energy-aware virtual machine consolidation in cloud data centers," in *Internet Technology and Secured Transactions (ICITST), 2017 12th International Conference for*. IEEE, 2017, pp. 61–66.

[31] G. Cybenko, "Dynamic load balancing for distributed memory multiprocessors," *Journal of parallel and distributed computing*, vol. 7, no. 2, pp. 279–301, 1989.

[32] T. C. K. Chou and J. A. Abraham, "Load balancing in distributed systems," *IEEE Transactions on Software Engineering*, no. 4, pp. 401–412, 1982.

[33] C. Zhao, S. Zhang, Q. Liu, J. Xie, and J. Hu, "Independent tasks scheduling based on genetic algorithm in cloud computing," in *Wireless Communications, Networking and Mobile Computing, 2009. WiCom'09. 5th International Conference on*. IEEE, 2009, pp. 1–4.

[34] M. A. Tawfeek, A. El-Sisi, A. E. Keshk, and F. A. Torkey, "Cloud task scheduling based on ant colony optimization," in *Computer Engineering & Systems (ICCES), 2013 8th International Conference on*. IEEE, 2013, pp. 64–69.

[35] F. Saffre, R. Tateson, J. Halloy, M. Shackleton, and J. L. Deneubourg, "Aggregation dynamics in overlay networks and their implications for self-organized distributed applications," *The Computer Journal*, vol. 52, no. 4, pp. 397–412, 2009.

[36] X. Ren, R. Lin, and H. Zou, "A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast," in *Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on*. IEEE, 2011, pp. 220–224.

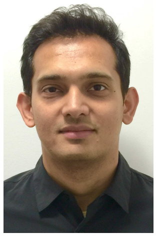
[37] M. Kuypers and E. Paté-Cornell, "Department of energy cyber security incidents," Center for International Security and Cooperation, Stanford, 2016.

[38] M. Smith and E. Paté-Cornell, "Cyber risk analysis for a smart grid: How smart is smart enough? a multi-armed bandit approach," in *Proceedings of the 2nd Singapore Cyber-Security R&D Conference (SG-CRC 2017)*, 2017, pp. 37–56.

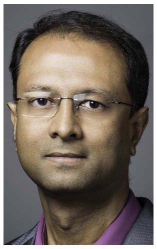
[39] B. Edwards, S. Hofmeyr, and S. Forrest, "Hype and heavy tails: A closer look at data breaches," *Journal of Cybersecurity*, vol. 2, no. 1, pp. 3–14, 2016.

[40] A. Gosavi, *Simulation Based Optimization: Parametric Optimization Techniques and Reinforcement Learning*. Norwell, MA: Kluwer Academic, 2003.

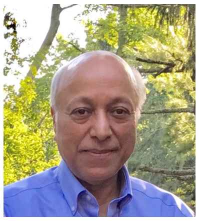
[41] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.



Ankit Shah received the B.S. degree in computer science from Florida Atlantic University, Boca Raton, FL, USA, in 2001, the M.S. degree in operations research from George Mason University, Fairfax, VA, USA, in 2016, and the interdisciplinary Ph.D. degree in information technology from George Mason University, Fairfax, VA, USA, in 2019. He is a Research Scientist at the Lawrence Livermore National Laboratory, CA, in the field of Reinforcement Learning. His research interests include cybersecurity analytics, decision-making under uncertainty, combinatorial optimization, stochastic dynamic programming, adversarial machine learning, and deep reinforcement learning.



Rajesh Ganesan received the M.S. degree in industrial engineering, the M.A. degree in mathematics, and the Ph.D. degree in industrial engineering from the University of South Florida, Tampa, FL, USA, in 2002, 2005, and 2005, respectively. He is currently an Associate Professor of systems engineering and operations research with George Mason University, Fairfax, VA, USA, where he is with the Center for Secure Information Systems and the Center for Air Transportation Systems Research. His research interests include stochastic optimization (approximate dynamic programming), Bigdata analytics, multiscale statistical data analysis using wavelets, and engineering education. His research applications include cybersecurity, healthcare, defense, air transportation, and nanomanufacturing. Dr. Ganesan is a Senior Member of the Institution of Incorporated Engineers and a member of the American Society for Engineering Education and INFORMS professional organization.



Sushil Jajodia (F'13) received the B.S. and M.S. degrees from Southern Illinois University, Carbondale, IL, USA, in 1969 and 1971, respectively, and Ph.D. degree from University of Oregon, Eugene, OR, USA, in 1977, all in mathematics. He is the University Professor, the BDM International Professor, and the founding Director of the Center for Secure Information Systems, Volgenau School of Engineering, George Mason University, Fairfax, VA, USA. He is also the Founding Site Director of the Center for Configuration Analytics and Automation (established under the National Science Foundation (NSF) Industry/University Cooperative Research Program), George Mason University. He held permanent positions at the NSF, Naval Research Laboratory, Washington, and the University of Missouri, Columbia. He has also been a Visiting Professor with the University of Milan, Italy; the Sapienza University of Rome, Italy; the Isaac Newton Institute for Mathematical Sciences, Cambridge University, U.K.; King's College London, U.K.; Paris Dauphine University, France; and Imperial College London, U.K. He has authored or coauthored seven books and more than 450 technical papers in the refereed journals and conference proceedings and edited 44 books and conference proceedings. He is also a holder of 21 patents. Prof. Jajodia was a recipient of the 1996 IFIP TC 11 Kristian Beckman Award, the 2000 Volgenau School of Engineering Outstanding Research Faculty Award, the 2008 ACM SIGSAC Outstanding Contributions Award, the 2011 IFIP WG 11.3 Outstanding Research Contributions Award, the 2015 ESORICS Outstanding Research Award, and the 2016 IEEE Computer Society Technical Achievement Award. He was recognized for the most accepted papers at the 30th anniversary of the IEEE Symposium on Security and Privacy. His h-index is 100 and Erdos number is 2.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2019.2927977, IEEE Transactions on Parallel and Distributed Systems

Pierangela Samarati (F'12) received her Ph.D. in Computer Science at Università degli Studi di Milano in 1988. She is a Professor at the Computer Science Department of the Università degli Studi di Milano. Her main research interests are in data protection, security, and privacy. She has participated in several projects involving different aspects of information protection. On these topics she has published more than 270 papers appeared in journals, conference proceedings, and books. She is co-author of the book ”Database Security,” Addison-Wesley, 1995. She has been Computer Scientist in the Computer Science Laboratory at SRI, CA (USA). She has been a visiting researcher at the Computer Science Department of Stanford University, CA (USA), and at the Center for Secure Information Systems, George Mason University, VA (USA). She is the chair of the IEEE Systems Council Technical Committee on Security and Privacy in Complex Information Systems (TCSPCIS), of the ERCIM Security and Trust Management Working Group (STM), and of the ACM Workshop on Privacy in the Electronic Society (WPES). She is the Coordinator of the Working Group on Security of the Italian Association for Information Processing (AICA), the Italian representative in the IFIP (International Federation for Information Processing) Technical Committee 11 (TC-11) on ”Security and Privacy”. She is a member of the Steering Committee of: European Symposium on Research in Computer Security (ESORICS), IEEE Conference on Communications and Network Security (CNS), IEEE Conference on Blockchain, Italian Conference on CyberSecurity, International Conference on Information Systems Security (ICISS), and International Conference on Information and Communications Security (ICICS). She has been named ACM Distinguished Scientist (2009) and IEEE Fellow (2012). She has received the ESORICS Outstanding Research award (2018), the IEEE Computer Society Technical Achievement Award (2016), the IFIP TC11 Kristian Beckman Award (2008), and the IFIP WG 11.3 Outstanding Research Contributions Award (2012).

Hasan Cam (SM'01) received the M.S. degree in computer science from Polytechnic University, New York, NY, USA, in 1986 and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 1992. He is a Computer Scientist at the U.S. Army Research Laboratory, Adelphi, MD, USA. He currently works on the projects involved with the development of cybersecurity metrics, models, and data analytics for assessing and managing cybervulnerability, risk, resilience, agility, mission assurance, and malware spreading over wired, mobile, and tactical networks. He serves as the Government Lead for the risk area in Cyber Collaborative Research Alliance. He has previously worked as a faculty member in the academia, and a Senior Research Scientist in the industry. He has served as an editorial member of two journals, a Guest Editor for two special issues of journals, an organizer of symposiums and workshops, and a Technical Program Committee member in numerous conferences. His research interests include cybersecurity, networks, algorithms, and parallel processing.