Optimal Scheduling of Cybersecurity Analysts for Minimizing Risk

RAJESH GANESAN, George Mason University
SUSHIL JAJODIA, George Mason University
HASAN CAM, Army Research Laboratory

Cybersecurity threats are on the rise with evermore digitization of the information that many day-to-day systems depend upon. The demand for cybersecurity analysts outpaces supply, which calls for optimal management of the analyst resource. Therefore, a key component of the cybersecurity defense system is the optimal scheduling of its analysts. Sensor data is analyzed by automatic processing systems, and alerts are generated. A portion of these alerts is considered to be *significant*, which requires thorough examination by a cybersecurity analyst. Risk, in this paper, is defined as the percentage of unanalyzed or not thoroughly analyzed alerts among the *significant* alerts by analysts. The paper presents a generalized optimization model for scheduling cybersecurity analysts to minimize risk (a.k.a maximize *significant* alert coverage by analysts) and maintain risk under a pre-determined upper bound. The paper tests the optimization model and its scalability on a set of given sensors with varying analyst experiences, alert generation rates, system constraints, and system requirements. Results indicate that the optimization model is scalable, and is capable of identifying both the right mix of analyst expertise in an organization and the sensor-to-analyst allocation in order to maintain risk below a given upper bound. Several meta-principles are presented, which are derived from the optimization model, and they further serve as guiding principles for hiring and scheduling cybersecurity analysts. The simulation studies (validation) of the optimization model outputs indicate that risk varies non-linearly with analyst/sensor ratio, and for a given analyst/sensor ratio, the risk is independent of the number of sensors in the system.

Categories and Subject Descriptors: C.2.0 Security and Protection [**D.4.6 Security and Protection**]: H.2.0 Security, Integrity, and Protection

General Terms: Cybersecurity, Analysts, Scheduling

Additional Key Words and Phrases: Cybersecurity analysts, optimization, resource allocation, risk mitigation, simulation, scheduling

ACM Reference Format:
Rajesh Ganesan, Sushil Jajodia, and Hasan Cam, 2015. Optimal Scheduling of Cybersecurity Analysts for Minimizing Risk. *ACM Trans. Intell. Syst. Technol.* x, x, Article xx (January 2015), 32 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Inspired by the pioneering works by Anderson [Anderson 1980] and Denning [Denning 1986; 1987], intrusion detection has been studied for over three decades. Much of the early research focused on misuse detection models (developing signatures of known at-

Rajesh Ganesan and Sushil Jajodia were partially supported by the Army Research Office under grants W911NF-13-1-0421, W911NF-15-1-0576, and W911NF-13-1-0317; and by the Office of Naval Research under grants N00014-13-1-0703 and N00014-15-1-2007. Author's addresses: R. Ganesan, Department of Systems Engineering and Operations Research, George Mason University; S. Jajodia, Center for Secure Information Systems, George Mason University, Fairfax, VA 22030-4422; H. Cam, Army Research Laboratory, 2800 Powder Mill Road, Adelphi, Maryland 20783-1138.

DOI: http://dx.doi.org/10.1145/0000000.0000000

tacks; any matched activity is flagged as an attack) or anomaly detection models (characterizing normal behaviors of all users and programs; any significant deviation from the normal profile is considered suspicious). Because the volume of alerts generated by intrusion detection sensors was overwhelming, a great deal of later work focused on developing techniques (based on machine learning [Sommer and Paxson 2010] or data mining [Barbara and Jajodia 2002], for example) for reducing false positives.

Recently, there has been some work that has focused on the need to improve efficiency of cybersecurity analysts [D'Amico and Whitley 2008; Erbacher and Hutchinson 2012; Zimmerman 2014]. The task of a cybersecurity analyst includes examining the alerts generated by an intrusion detection system (IDS) such as SNORT or a Security Information and Event Management (SIEM) tool such as Bro [Paxson 1999] and identifying those that are considered as significant and require further investigation. Table I provides an example of cyber incident and event categories used by the US Department of Defense [Chief Information Officer 2008]. All alerts are categorized on a scale of 1-9 with category 1, 2, 4, and 7 (incidents) being severe ones. Category 1, 2, 4, and 7 alerts are analyzed and reported to a watch officer, and then a report has to be written. See Figure 1.

From an alert investigation point of view, there are two metrics that impact a cybersecurity organization - one of them is a quantity metric that measures the number of alerts thoroughly investigated among those generated, which highlights the capacity of the organization, and the other is the quality metric that measures the accuracy of investigation, which highlights the true/false positive and true/false negative rates of the alert investigation process. The scope of the paper is focused on the first metric (quantity) of determining whether or not an organization has the capacity to analyze all the alerts and the significant alerts within them based on available resources, time constraints, manpower constraints, and shift scheduling constraints. The premise of the paper is that an organization must have the capacity to investigate all the alerts generated in a day, otherwise the unanalyzed or not thoroughly analyzed alerts will pose a threat to the organization. It is true that unless an alert is thoroughly analyzed, its category or severity is unknown. Also, the time taken to analyze an alert depends on its category or severity, whether or not it is a known or a new pattern of alert, and the expertise level of the analyst. Therefore, at the time of drawing an alert from the queue for investigation, since its category or severity is unknown, the time to analyze an alert in this paper is based upon an average time from a probability distribution, which can be obtained from historical real world data. The total time needed to thoroughly analyze all the alerts and significant alerts can be compared to the total time available, which is based on the current capacity of the organization and sensor-to-analyst allocation rules, in order to determine the % of significant alerts that would remain unanalyzed (defined as risk for this paper). Such a risk metric could be used to initiate actions to build analyst capacity for the organization with optimal sensor-to-analyst allocation and optimal shift schedules. Hence, the scope of the paper is focused on capacity building for a cyber-defense organization through the optimal allocation and scheduling of its analysts, regardless of the type of alert (category or severity), using the notion that some alerts will need more time than the others. Several factors are considered in this paper to calculate the alert investigating capacity of the organization, which includes number of sensors, an average alert generation rate for the sensors, number of analysts, their expertise level, sensor-to-analyst allocation, analyst time to investigate an alert, and their work-shift schedule. The paper assumes that all the alerts that were thoroughly investigated were also accurately categorized. It should be noted that as a second metric (quality), once a significant alert has been detected by thorough alert analysis, a different definition of risk can be used to measure the quality of work performed by capturing the true positive and false negative rates.

Table I. Alert Categories [Chief Information Officer 2008]

Cat 1-9	Description
1	Root Level Intrusion (Incident): Unauthorized privileged access (administrative or root access) to a DoD system.
2	User Level Intrusion (Incident): Unauthorized non-privileged access (user-level permissions) to a DoD system. Automated tools, targeted exploits, or self-propagating malicious logic may also attain these privileges.
3	Unsuccessful Activity Attempted (Event): Attempt to gain unauthorized access to the system, which is defeated by normal defensive mechanisms. Attempt fails to gain access to the system (i.e., attacker attempt valid or potentially valid username and password combinations) and the activity cannot be characterized as exploratory scanning. Can include reporting of quarantined malicious code.
4	Denial of Service (DOS) (Incident): Activity that impairs, impedes, or halts normal functionality of a system or network.
5	Non-Compliance Activity (Event): This category is used for activity that, due to DoD actions (either configuration or usage) makes DoD systems potentially vulnerable (e.g., missing security patches, connections across security domains, installation of vulnerable applications, etc.). In all cases, this category is not used if an actual compromise has occurred. Information that fits this category is the result of non-compliant or improper configuration changes or handling by authorized users.
6	Reconnaissance (Event): An activity (scan/probe) that seeks to identify a computer, an open port, an open service, or any combination for later exploit. This activity does not directly result in a compromise.
7	Malicious Logic (Incident): Installation of malicious software (e.g., trojan, backdoor, virus, or worm).
8	Investigating (Event): Events that are potentially malicious or anomalous activity deemed suspicious and warrants, or is undergoing, further review. No event will be closed out as a Category 8. Category 8 will be re-categorized to appropriate Category 1-7 or 9 prior to closure.
9	Explained Anomaly (Event): Events that are initially suspected as being malicious but after investigation are determined not to fit the criteria for any of the other categories (e.g., system malfunction or false positive).

Furthermore, the severity of the threat that an alert poses to the organization, and actions to mitigate the threat can be taken. However, such as definition of risk and the actions to mitigate are beyond the scope of this paper and are mentioned as part of future work. The notion of risk used in the paper is defined next.

We use a precise notion of risk which is defined as the percentage of the significant alerts that were not thoroughly or properly analyzed. This insufficient analysis of alerts could happen due to the following reasons such as sub-optimal scheduling of analysts, not enough personnel to analyze, lack of time to analyze a significant alert, and/or not having the right mix of expertise in the shift in which the alert occurs. The objective of the paper is to determine the analyst scheduling strategy that minimizes the above risk and maintains risk under a pre-determined upper bound for a set of system defined parameters and constraints. The scheduling strategy includes the determination of the sensor-to-analyst allocation and the desired expertise mix of the analyst workforce. The paper considers three levels of analysts based on their work experience - senior L3, intermediate L2, and junior L1 level analysts. An analyst can be allocated to more than one sensor based on their experience. Finally, the time taken to analyze an alert is dependent on the experience of the analysts.

In this paper, we view cybersecurity analysts as a resource that must be allocated to the process of examining alerts in an optimal way that minimizes risk while meeting the resource constraints. Such resource constraints include the number of sensors on which an analyst can be trained or assigned, the expertise mix that the cybersecurity defense organization wishes to have, the time taken by an analyst to investigate an alert (translates to analyst workload), and preferences such as shift hours and days-off in a week of the analysts.

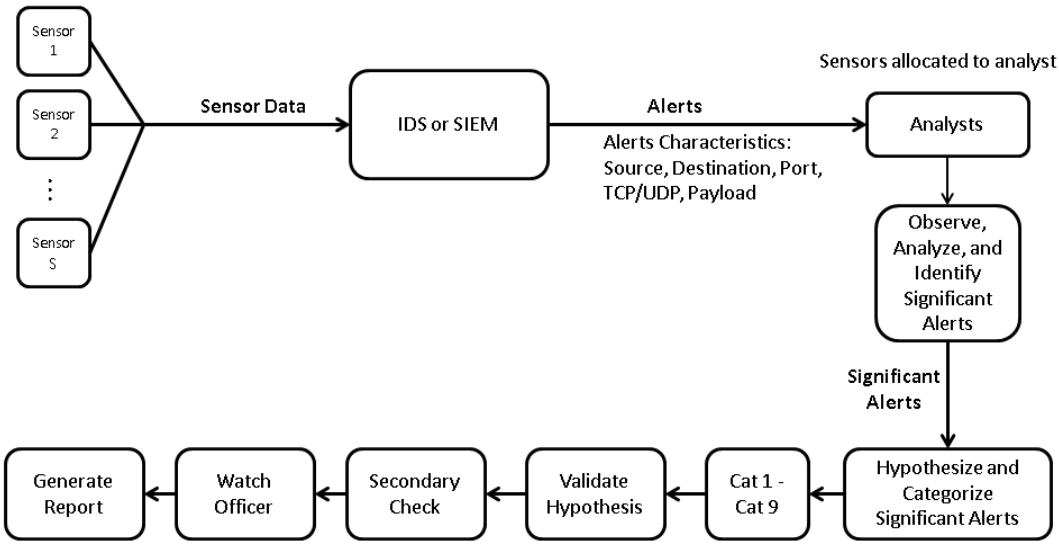

Fig. 1. Alert Analysis Process.

There are several contributions in this paper. The first contribution of the paper is the mixed-integer programming based optimization algorithm that determines both the optimal sensor-to-analyst allocation and the right mix of analyst experience to minimize risk and keep risk below a given pre-set level that is desirable for the cybersecurity system. The above risk minimization capability is a direct measure of the goodness of the analyst resource allocation system. It should be noted that the optimization algorithm is a generic one and is independent of the number of sensors, however, the algorithm's output adapts to the number of sensors in the system. The optimization model is solved using genetic algorithm heuristics. The second contribution is a stand-alone simulation framework that can be used to simulate a given sensor-to-analyst allocation strategy and to validate the outcomes of the optimization model. The third contribution is a novel algorithm for days-off scheduling that can schedule the cybersecurity analysts based on their shift-hours and days-off preferences. The fourth contribution is to introduce the relationship between risk and analyst/sensor ratio, which is very useful in making analyst hiring decisions. The other contributions include a set of meta-principles that provide the general guidelines for developing and implementing an efficient cybersecurity analyst system.

The paper is organized as follows. In Section 2, a motivational background on alert analysis is provided, which includes the current alert analysis process, the categorization of alerts, and our definition of risk for this paper. Section 3 presents the related literature in cybersecurity analyst scheduling. In Section 4, the model parameters and assumptions that affect cybersecurity analyst scheduling are presented. The model parameters include fixed parameters, systems requirement parameters, decision parameters. Section 5 presents the sensor-to-analyst resource allocation model consisting of an optimization model, a scheduler model, and a simulation model. Individual algorithms for each model have been developed and presented. Computational complexity is also discussed. Section 6 presents the results from testing all of the above models. Finally, in Section 7, the conclusions and future directions of this research are presented.

2. BACKGROUND ON ALERT ANALYSIS AND OUR DEFINITION OF RISK

In this section, a background of the current alert analysis process and its categorization are presented. Also the definition of risk is provided.

2.1. Current Alert Analysis Process

Alerts are generated and analyzed by cyber security analysts as shown in Figure 1. The network sensor data collected from the internet is analyzed by an IDS or a SIEM, which automatically analyses the data and generates alerts. Most of the alerts are deemed insignificant by the above centers, and about 1% of the alerts generated are classified as significant alerts.[1] The significant alerts are those with a different pattern in comparison to previously known alerts. The significant alerts must be further investigated by cybersecurity analysts and categorized.

All alerts have their sensor identification information as one of its characteristics. Also, sensors are identified and allocated to analyst identity. The scheduling algorithm, specifically allocates analysts to a work shift in a 14-day period. Besides, watch-officers are responsible for presiding over the shift (ensuring sensor to analyst allocation, looking for absentees, looking for unauthorized personnel, and monitoring/overseeing the shift activities). Any unusual activity, such as resource access outside business hours, will trigger an alert, which must be analyzed.

Alert correlation is an important topic and pertains to a collection of IDS or SIEM alerts [Valeur et al. 2004]. Correlation is done by SIEM, or automatically by analyst tools, or manually by analysts. Alert correlation logic is built over the output from disparate log sources. Similarity based clustering or causal relationship based reasoning is often used for alert correlation, and uncertainty is captured via Bayesian networks. Yet, the task of alert correlation is made challenging by the ever changing threat environment and by the need for scalability to handle large complex coordinated attacks. For example, alert correlation can use various security events such as unusual port activities in firewall (malicious scanning), Apache server exploit, suspicious DNS requests, logs from web application firewall, threats recognized from antivirus, transmission of sensitive data in plain text, and resource access outside business hours to detect a potential threat. Correlation can capture a high level view of the attack activity on the target network without losing security-relevant information. The output of alert correlation is the grouping of alerts, which are then investigated [Du and Yang 2013] by analysts. A more detailed example of alert correlation can be found in [Sadoddin and Ghorbani 2006].

In the current system, sensors are pre-assigned to analysts, who investigate the alerts generated from them. If alert correlation is performed then the grouped alerts from various heterogeneous sensors are presented to the analysts that are assigned to those sensors. The type and the number of sensors allocated to an analyst depend upon the experience level of the analysts. The experience level of an analyst further determines the amount of workload that they can handle in an operating shift. The workload for an analyst is captured in terms of the number of alerts/hr that can be analyzed based on the average time taken to analyze an alert. When alerts are grouped, the total time taken to analyze is the sum of the individual time taken to analyze the alerts within the group. For simplicity, in this paper, we assume that a set of grouped alerts will be analyzed by only one analyst so that the number of alerts and the time spent for analysis can be measured. If due to lack of expertise, a second analyst is added to the same group of alerts, then the alerts investigated by the two analysts

[1] We arrived at the 1% figure based on our literature search [Julisch and Dacier 2002; Goodall et al. 2004] and numerous conversations with cybersecurity analysts and Cybersecurity Operations Center (SOC) managers. Our model treats this value as a parameter that can be changed as needed.

will be distinct. In other words, the paper assumes that an alert (whether or not it is grouped with others) is investigated by only one analyst. In this paper, three types of analysts are considered (senior L3, intermediate L2, and junior L1 level analysts), and their workload values are described later under model parameters.

The cybersecurity analysts must observe the significant alerts that are classified by the IDS or SIEM, and analyze those significant alerts that are pertinent to their pre-assigned sensors. An important output of this stage of analysis by the analysts is the categorization of the alerts, which is explained next.

2.2. Alert Categorization

A cybersecurity analyst must do the following: 1) analyze the alerts and identify the significant alerts, and 2) hypothesize the severity of threat posed by a significant alert and categorize the significant alert under Category 1-9. The description of the categories are given in Table I [Chief Information Officer 2008]. If an alert is hypothesized as a very severe threat and categorized under Cat 1, 2, 4, or 7 (incidents) then the watch officer for the shift is alerted and a report is generated (see Figure 1). For this paper, it is assumed that all the alerts that were thoroughly investigated were also accurately categorized. Therefore, false positives and false negatives in alert investigation are to be modeled separately as part of future work.

2.3. Our Definition of Risk

Cybersecurity risk can be interpreted in many ways. A successful intrusion can be called a risk and a count on the number of such successful intrusions can be used to quantify risk. Severity of risk (from true positives and false negatives) can also be measured in terms of damages caused by an intrusion. The number of significant alerts that are categorized under Cat 1 and 2 can be used as a metric to quantify risk. There isn't a single notion of risk that fits all research questions in the field of cybersecurity. In this paper, alert coverage and risk are defined as follows, which is most appropriate for the optimal scheduling problem. Alert coverage is defined as the % of the significant alerts identified by the IDS or SIEM that are thoroughly investigated in a work-shift by analysts and the remainder that is not properly analyzed constitutes the risk. It is to be reminded that the significant alerts constitute about 1% of all alerts issued by the IDS or SIEM after analyzing sensor data. Therefore, the objective of a cyber-defense organization is to minimize risk, which is to minimize the number of significant alerts that were not properly analyzed or unanalyzed. Risk can be stated as follows:

$$Risk\% = 100 - Alert\ Coverage\% \tag{1}$$

2.3.1. Computational Complexity and Risk as an Upper Bound. In the following section, the paper explains that for the optimization model, risk is used as a constraint with an upper bound instead of being a direct objective of the optimization algorithm. In this research, risk is proportional to a combination of factors, including the number of analysts, their experience mix, analyst's shift and days-off scheduling, and their sensor assignment in a work shift. Hence, scheduling cybersecurity analysts is a critical part of cybersecurity defense. For the cybersecurity alert analysis process described above, it is imperative to have an efficient cybersecurity analyst scheduling system, which in turn will minimize risk. The requirements of such a system can be broadly described as follows. The cybersecurity analyst scheduling system,

(1) shall ensure that risks due to threats are maintained at the minimum,
(2) shall ensure that an optimal number of staff is available and are optimally allocated to sensors to meet the demand to analyze alerts,

(3) shall ensure that a right mix of analysts are staffed at any given point in time, and
(4) shall ensure that weekday, weekend, and holiday schedules are drawn such that it conforms to the working hours/leave policy of the organization.

The optimization model uses a mixed-integer programming formulation. Integer programming problems belong in general to the NP-complete class because the feasible region is not a convex set. The closest known algorithms for our problem are a quadratic Assignment problem (QAP), Generalized Assignment Problem (GAP) or a Bipartite Matching Problem which are known to be NP-hard. In those problems, every job is assigned to only 1 agent. In our case, it is many to many, which is even harder than GAP. Since there is no known provably efficient algorithm to reduce our problem polynomially and the closest ones given above are NP-hard, our problem is at least NP-hard. For example, if there are i sensors and j analysts, and multiple sensors can be assigned to an analyst and *vice versa*, the complexity is of the order $O(2^{i*j})$. Since the number of discretizations of the feasible solution space can lead to a large number of solutions, the sensor-to-analyst optimization problem is NP-hard to be solved optimally. Even if some allocations are infeasible, it is still computationally impractical to evaluate the remaining feasible allocations and determine the optimal allocation of sensors to an analyst. The mixed-integer optimization problem with risk minimization as the direct objective will continue to search the solution space for the best solution that has the lowest risk by starting the search from an initial solution, which in turn would become computationally expensive. Furthermore, while searching for a solution that has the lowest risk, there is a high potential to increase the number of analysts, which could become prohibitively expensive for the organization. Also, the direct minimization of risk approach is not scalable as the number of sensors and analysts are increased.

Metaheuristics provide an important alternative to solve the NP-hard class of problems [Talbi 2009]. By setting an upper bound for risk, the optimization problem can be solved faster using metaheuristic techniques such as genetic algorithm. Using a heuristic search, several solutions can be initially tested to determine a set of feasible solutions that meet the risk upper bound. Then the search can both diversify and intensify iteratively to obtain several sets of feasible solutions that have lower number of analysts, and among those feasible solutions the algorithm will choose the ones with a lower risk than the previously obtained solutions while keeping the risk below the pre-set upper bound. Thus, the optimization will minimize both the total number of analysts in the organization and risk subject to the constraints imposed by the risk upper bound, analyst utilization, and those by the system resources. The above heuristics approach is computationally both viable and faster, and the search can be terminated using a stopping criteria. It should be noted that optimality can be proved with NP-hard problems solved using heuristics, however, good-enough solutions can be obtained.

The relationship between risk and available resource (cybersecurity analysts) can be understood from Figure 2. The trivial regions are those where either there is too much resource (very low risk) or too little resource (very high risk) wherein any amount of optimal scheduling will not significantly affect the risk levels. Optimal scheduling gains importance only when there is just enough resource that must be managed well for reaching the lowest possible risk. Two types of models can be studied from Figure 2. First, given a certain available cybersecurity analyst resource (*i.e.* number of analysts, experience mix, and sensor-to-analyst allocation), a *simulation* model can be constructed to evaluate the risk for a given significant alert generation rate. This is depicted from $C \rightarrow D$. However, several simulations will be needed to measure the risk associated with different combinations of the available cybersecurity analyst resource

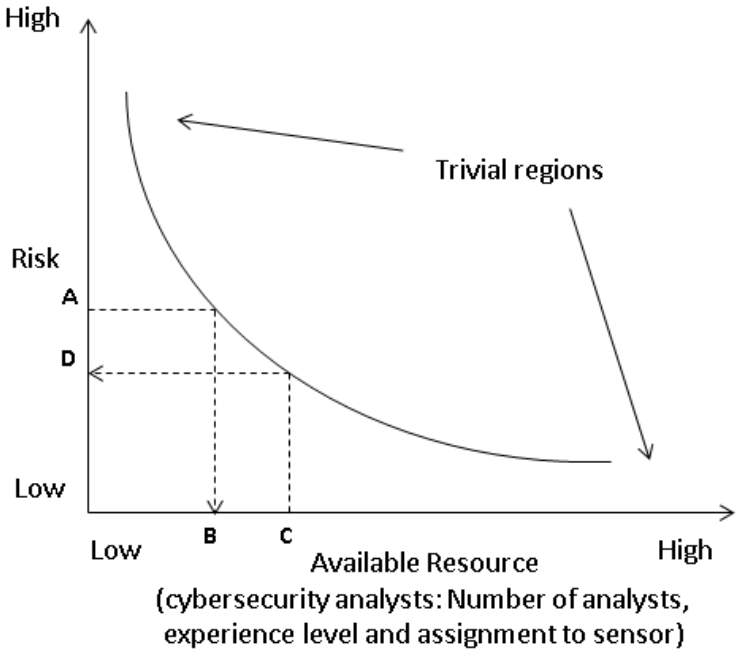

Fig. 2. Risk vs Available Resource

by varying the number of analysts, experience mix, and sensor-to-analyst allocation. Such simulations are computationally impractical to implement on a large-scale to determine the best sensor-to-analyst allocation and the right analyst expertise mix. The second model is an *optimization* model in which an optimal cybersecurity analyst schedule is obtained after determining the optimal number of analyst, right experience mix, and optimal sensor-to analyst allocation for a given upper bound on the risk level and for a given significant alert generation rate. This is depicted from $A \rightarrow B$ in Figure 2. The output of the optimization model can then be used to generate the shift and days-off schedule for the analysts.

In summary, the paper presents an optimization algorithm framework that meets the above system requirements and delivers an optimal cybersecurity analyst schedule that maintains risk at or below a certain pre-determined risk level under a given set of system imposed parameters and constraints. The optimal cybersecurity analyst schedule is drawn after the optimization algorithm determines the number of analyst, experience mix, and sensor-to analyst allocation. Next, the paper presents some of the related literature in cybersecurity analyst scheduling.

3. RELATED LITERATURE

Intrusion detection has been an active area of research for over three decades [Northcutt and Novak 2002; Di Pietro and Mancini 2008]. Much of the work has focused on developing automated techniques for detecting malicious behavior. Because the number of alerts generated by intrusion detection sensors is very large, a great deal of effort has been devoted to developing automated alert reduction techniques. Indeed, there are open source [Paxson 1999] and commercially available [Zimmerman 2014] Security Information and Event Management (SIEM) tools that take the raw sensor

data as input, aggregate and correlate them, and produce alerts that require remediation by cybersecurity analysts.

The paper differs from the above literature by taking the position that cybersecurity analysts should also be viewed as a resource that must be allocated to sensors in an optimal way that minimizes risk while meeting the resource constraints. It develops a generic optimization algorithm that provides the flexibility to schedule the cybersecurity analysts for a given number of sensors, an experience mix that the organization desires, and a pre-set risk level that serves as an upper bound for risk mitigation. The algorithm will allocate L1, L2 and L3 analysts to sensors with L3 analysts being assigned to more sensors than L2 analysts, and L2 analysts assigned to more sensors than L1 analysts. Furthermore, the generic nature of the simulation model provides the flexibility to perform several sensitivity analyses of its parameters such as varying the alert generation rate, the experience mix that is desired, the number of sensors, and time taken to analyze the alerts for different experience categories of the analysts.

4. MODEL PARAMETERS

This section presents the details of the model parameters of the cybersecurity analyst resource allocation model, which is presented in the next section. In the following, the fixed input parameters, system-requirement parameters, decision parameters, and underlying assumptions of the resource allocation model are presented.

4.1. Fixed Parameters

Several input parameters are maintained fixed for the optimization model. The number of sensors, average alert generation rate from the sensors, which follows a Poisson or uniform distribution, and analyst characteristics are kept fixed during the execution of the optimization model. The analyst characteristics consists of the following components: analyst availability (binary indicator variable), analyst experience level, number of sensors allocated to an analyst, which depends on their experience, and time taken to analyze an alert for a given level of analyst experience. The paper uses three levels of analyst experience as given below.

(1) L3- senior analyst. L3 analysts are assigned 4-5 sensors and they and can handle on average 12 alerts per hour (5 min/alert).
(2) L2- intermediate analyst. L2 analysts are assigned 2-3 sensors and they can handle on average 7-8 alerts per hour (8 min/alert).
(3) L1- junior analyst. L1 analysts are assigned 1-2 sensors and they and can handle on average 5 alerts per hour (12 min/alert).

4.2. System-Requirement Parameters

The cybersecurity analyst allocation model must meet the following system requirements.

(1) The system must minimize risk (maximize alert coverage by analysts). For the optimization model, an upper bound on risk is specified as a constraint.
(2) Analyst utilization must exceed 95%, which is added as a constraint to the optimization model. This is computed based on the amount of time an analyst is idle during the shift. This allows the optimization algorithm to allocate enough sensors to analysts based on their experience level (L1, L2, or L3) and alert generation rates, such that they are all highly utilized.
(3) The desired mix of expertise level among analysts for an organization must be specified such as 20-40% L1, 30-50% L2, and 30-40% L3 level analysts to facilitate training of juniors and providing career opportunities such as promotions within

an organization. This will prevent the optimization model to pick all L3 level analysts who are the most efficient in examining the alerts.

4.3. Decision Parameters

The decision parameter of the optimization model is the analyst-to-sensor allocation that satisfies the system-requirement and adheres to the characteristics set by the fixed parameters of the model. The assignment can be envisioned as a matrix of 0's and 1's where the rows represent analyst identity and columns represent sensor identity (see Tables IV-VI). Also, '1' means that a sensor has been assigned to an analyst, and '0' otherwise. From this allocation matrix, the number of analysts at each level of expertise that are needed per day can be derived.

4.4. Model Assumptions

The assumptions of the optimization model are as follows.

(1) All alerts that were thoroughly investigated were also accurately categorized. Hence, false positives and false negatives are not modeled in this paper.
(2) An analyst's average service time to investigate an alert can either remain fixed as given under the fixed parameters above or be allowed to change by drawing from a distribution such as Poisson or Uniform.
(3) Analysts work in two 12-hr shifts, 7-AM-7PM and 7PM-7AM. However, the optimization model can be adapted to 8 hr shifts as well.
(4) At the end of the shift all analysts must complete the last alert that they are investigating even if there is a small spillover of their time into the next shift. This could result in analyst utilization to exceed 100%. However, no new alert will be taken up by an analyst once their shift time is completed.
(5) When a group of analysts are allocated to a group of sensors by the optimization algorithm, the alerts generated by that group of sensors are arranged in a single queue based on their arrival time-stamp, and the next available analyst within that group will draw the alerts from the queue based on a first-in-first-out rule.
(6) Based on experience, an analyst spends, on average, about the same amount of time to investigate alerts from the different sensors that are allocated, which can be kept fixed or drawn from a probability distribution.
(7) Analysts of different experience levels can be paired to work on a sensor.
(8) Significant alerts are generated from a distribution such as Poisson or Uniform. The paper uses an average alert generation rate, and on some days the alert rate will be above average and on others below average. The average number of alerts generated per sensor is kept equal and fixed throughout the day for all sensors. However, the model can be adapted easily to the situation where the average number of alerts generated per sensor is unequal but fixed throughout the day. It should be noted that as alert rates increase beyond the analyzing capacity of the work shift then the risk will also increase for that shift. For a dynamic alert generation rate that changes by the hour, a control-theoretic approach to optimization with future workload prediction is required, which is one of the future works of this research.
(9) The optimization model is run for 24-hrs to determine the sensor-to-analyst allocation for that day. Simulation statistics on risk and analyst utilization are calculated at the end of the 24-hr day.
(10) Writing reports of incidents and events during shifts is considered as part of alert examining work, and the average time to examine the alert includes the time to write the report.

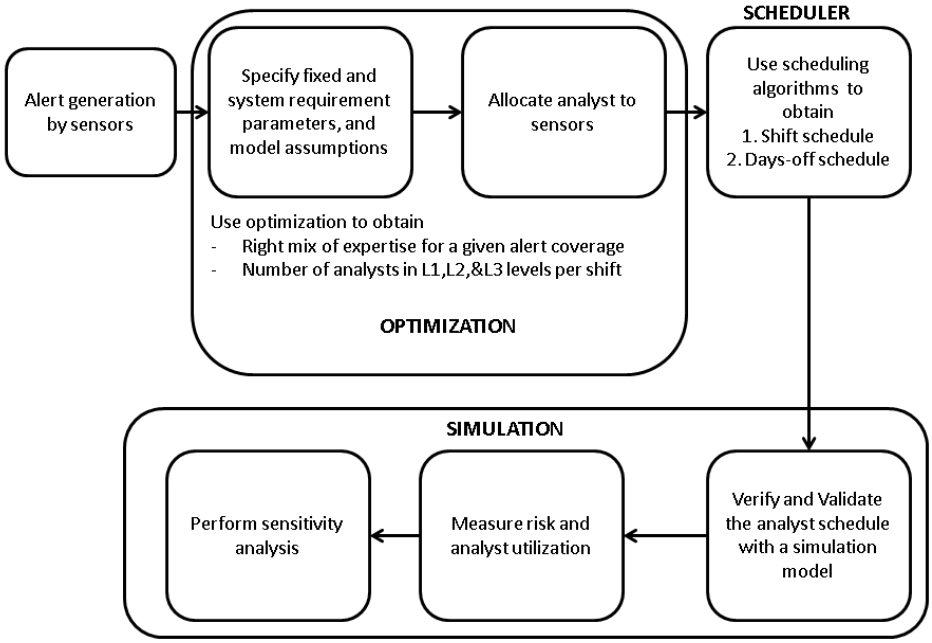

Fig. 3. Analyst resource allocation model.

5. CYBERSECURITY ANALYST RESOURCE ALLOCATION MODEL

This section presents the details of the cybersecurity analyst resource allocation model. The framework of the model is provided in Figure 3. The optimization model is run for a 24-hr period over several iterations to determine the best solution for sensor-to-analyst allocation and the right mix of analyst expertise under a given set of model parameters and system constraints. The analyst resource allocation model consists of three modules: the optimization, the scheduler and the simulation module.

5.1. Optimization Module

The optimization problem is modeled as a mixed-integer programming algorithm and is solved using genetic algorithm heuristics [Winston 2003], [Goldberg 1989], [Holland 1975]. The optimization module receives the fixed inputs and system requirements as specified in the previous section. The objective of the optimization model is to minimize both the number of analysts needed and risk subject to the constraints on analyst utilization, upper bound on risk, system constraints on analyst workload, and the desired experience level mix that is specified by the organization. The optimization algorithm begins with the assumption that a large number of analysts are available at each experience level, among which the minimum number will be selected such that the constraints are met. As explained earlier, the solution to the optimization model using heuristics search approach will seek the best solution that has the minimum number of analysts, and among several feasible solutions it will choose the one that has smallest risk under the given risk upper bound.

Define $n \in N$, where N is the number of sensors in the system, and n is the sensor identity. Let A, B, C represent the number of analysts of type L1, L2 and L3 respectively that are needed in a 24 hr period (1 day) such that their upper bound is given by $max(A) = max(B) = max(C) = N$, and $a \in A$, $b \in B$, $c \in C$ where $\{a, b, c\}$ represent an-

alyst identity. The upper bound assumes that the maximum number of analysts in any expertise category will be N. The upper bound serves as an input parameter that can be adjusted, and is used to initiate the algorithm for solving the optimization. Define $x_{a,n} = 1$ if analyst a from type L1 is allocated to sensor n, and 0 otherwise. Similarly, define $x_{b,n}$ and $x_{c,n}$ for analyst type L2 and L3 respectively. Also, define $x_a = 1$ if analyst a from type L1 is allocated to *any* sensor n, and 0 otherwise. Similarly, define x_b and x_c for analyst type L2 and L3 respectively.

The objective function can be defined as follows

$$Z = Min \ \{A + B + C\} = Min \ \left(\sum_{a=1}^{A} x_a + \sum_{b=1}^{B} x_b + \sum_{c=1}^{C} x_c \right) \tag{2}$$

where Z is the total number of analysts needed per day in the organization, and

$$x_a \ = \ 1 \ \ if \ \ \sum_{n=1}^{N} x_{a,n} \geq 1 \quad \forall a \tag{3}$$

$$= \ 0 \ \ otherwise \tag{4}$$

The above equation can be written in a linear form as follows

$$\sum_{n=1}^{N} x_{a,n} \leq N x_a. \tag{5}$$

Similar equations are written for x_b and x_c.

The constraints on the number of sensors allocated to an analyst are given as follows. Define L_a and U_a as the lower and upper bound for the number of sensors that can be allocated to type L1 analyst. Then

$$L_a \leq \sum_{n=1}^{N} x_{a,n} \leq U_a \ \ \forall a \tag{6}$$

Similar constraints as in (6) are written for L2 and L3 analysts respectively.

The constraint on the proportion of analysts with different expertise level in an organization is modeled as follows. Define L_A and U_A as the lower and upper bound for the proportion of L1 type analyst in the organization. Then it follows that

$$L_A \left(\sum_{a=1}^{A} x_a + \sum_{b=1}^{B} x_b + \sum_{c=1}^{C} x_c \right) \leq \sum_{a=1}^{A} x_a \leq U_A \left(\sum_{a=1}^{A} x_a + \sum_{b=1}^{B} x_b + \sum_{c=1}^{C} x_c \right) \ \ \forall a \tag{7}$$

Similar constraints as in (7) are written for L2 using L_B and U_B, and for L3 using L_C and U_C analysts respectively.

Define M as the total number of significant alerts generated by N sensors. Define p as the upper bound on risk (%) that the cybersecurity system can tolerate, where risk % is defined as the proportion of significant alerts that were unanalyzed or not thoroughly analyzed. Therefore, $p = 0\%$ means that all the significant alerts must be thoroughly investigated.

Define t_a, t_b, and t_c as the average time (in hours) taken by an analyst with L1, L2, and L3 level experience respectively to analyze an alert. Define $m_{a,n}$, $m_{b,n}$, and $m_{c,n}$ be the number of significant alerts analyzed by analyst $\{a, b, c\}$ respectively who are allocated to sensor n. Then it follows that

$$\sum_{a=1}^{A} \sum_{n=1}^{N} m_{a,n} + \sum_{b=1}^{B} \sum_{n=1}^{N} m_{b,n} + \sum_{c=1}^{C} \sum_{n=1}^{N} m_{c,n} \geq (1 - \frac{p}{100}) M \tag{8}$$

Analysts are turned on '1' or off '0' with a 0-1 variable $\mathcal{I}_a$ to indicate whether they are available in a shift or not. When the shift is over their status is turned off so that no new alerts are analyzed by them. Also with a 0-1 binary variable an analyst status in a shift is indicated as busy '1' and idle '0'. An idle status occurs when the queue of alerts waiting for investigation is empty. If the queue is nonempty then the analyst will immediately pull an alert from the queue for investigation as soon as the investigation of the current alert is completed. As per the model assumptions, since an analyst of a certain experience level spends, on average, the same amount of time on any alert from any sensor that is allocated to them, the total time spent by an analyst a is given by

$$T_a = \mathcal{I}_a t_a x_a \sum_{n=1}^{N} m_{a,n} \quad \forall a \tag{9}$$

If the average time (in hours) taken by an analyst with L1, L2, and L3 level experience respectively to analyze an alert varies by sensor n then the total time spent by an analyst a is given by

$$T_a = \mathcal{I}_a x_a \sum_{n=1}^{N} t_{a,n} m_{a,n} \quad \forall a \tag{10}$$

where $t_{a,n}$ is the average time taken by analyst type L1 to investigate alert from sensor n.

Consequently in a 24 hr day, the analyst utilization ($\geq 95\%$) constraint is given by

$$T_a/24 \geq 0.95 \quad \forall a \tag{11}$$

Similar constraints as in Equation (9) or (10), and (11) are written for L2 and L3 analysts respectively. It should be noted that all calculations are done per day (24-hr period). Since analysts work in 12-hr shifts as per the model assumptions, the optimization solution for the number analysts will be doubled to get the required number of analysts for a day.

The main solution output from the optimization problem is the sensor-to-analyst allocation for L1, L2, and L3 type analyst, $x_{a,n}$, $x_{b,n}$, and $x_{c,n}$ respectively. From this solution, the number of analysts with different experience levels to be hired in a day can be derived using Equation (3)-(5). The main solution vector from the optimization algorithm is a string of binary values, which is represented as follows

$$\mu = \{x_{a,n}, \ldots, x_{A,N}, x_{b,n}, \ldots, x_{B,N}, x_{c,n}, \ldots, x_{C,N}\}, \tag{12}$$

where

$$a = \{1, \ldots, A\}, b = \{1, \ldots, B\}, c = \{1, \ldots, C\}, n = \{1, \ldots, N\}, \{x_{a,n}, x_{b,n}, x_{c,n}\} \in (0, 1). \tag{13}$$

μ can also be represented in a matrix form in the final output with sensor identity as columns and analyst identity as rows (see Tables IV-VI).

For large numbers of sensors and analysts, solving the mixed-integer programming formulation using techniques such as brand-and-bound and cutting-plane methods is computationally prohibitive due to the NP-hard nature of the problem. Hence, the optimization problem to determine a good-enough feasible sensor-to-analyst allocation that meets the constraints of the model is solved using genetic algorithm - a well-known population metaheuristics technique. Several initial feasible solutions μ (as in Equation 12) are generated and evaluated, and the genetic algorithm would

further generate and evaluate new solutions (off-springs) using crossover and mutation[Goldberg 1989]. The algorithm applies k-point or uniform crossover to two solutions and a mutation (bit flipping) to randomly modify the individual solution contents to promote diversity. The crossovers and mutations are done independently for L1, L2, and L3 level analysts (different segments of Equation (12) that refers to each analyst level). After evaluating the objective function, the most promising subset of parents and offsprings are carried forward into the next generation (iteration). Replacement is done wherein the off-springs systematically replace the parents. For solutions with identical objective function (total number of analyst in each expertise level) but a different sensor-to-analyst allocation, the tie is broken by choosing solutions that have the lowest risk. The above not only provides feasible solutions that keep risk under the given upper bound but also provides the best allocation (found so far) that has the lowest risk. The genetic algorithm is terminated (stopping criteria) after a certain number of iterations show no improvement over the existing best solution that has been found. The steps of the optimization algorithm are given in Algorithm 1.

5.2. Scheduler Module

The input to the scheduling module is the number of L1, L2, and L3 level personnel needed per day $\{A, B, C\}$, which is derived from the sensor-to-analyst allocation per day. It is to be noted that the optimization algorithm is solved over a 24 hr period and the values of $\{A, B, C\}$ are the number of analysts of type L1, L2, and L3 respectively needed per day. A feasible days-off scheduling can be derived based on the following constraints.

(1) Each analyst gets at least 2 days off in a week and every other weekend off.
(2) An analyst works no more than 5 consecutive days.
(3) An analyst works 80 hrs per two weeks counted over 14 consecutive days between a Sunday and a Saturday. Both 12 hr and 8 hr shift patterns are allowed.

The objective of the scheduler algorithm is to find the minimum number of employees that must be in the organization in order to meet the daily demand and personnel work schedule preferences. A cyclic days-off staffing algorithm is presented below in Algorithm 2. Several other workforce scheduling algorithms are available in the literature and the details of the cyclic days-off staffing algorithm that is used in this paper are available in [Pinedo 2009]. The inputs to the cyclic days-off staffing algorithm are 1) types of days-off schedules, which is represented as the matrix Y given below, and 2) staffing requirements per day $\{A, B, C\}$ for each type of L1, L2, and L3 level employee respectively, which is obtained from the optimization module.

For a 12 hr-shift per day, the Y matrix that satisfies the above constraints is given in Table II. In the Y table, '0' represents days-off and '1' for days-on. For a 12 hr-shift per day, an employee has to work for 7 days in a 14 day period from a Sunday to Saturday with one of the 7 days for only 8 hours. Each column in the table represents a daily pattern of on and off days. For example, in Y, columns 1 and 2 represent first and third weekend off, whereas columns 3 and 4 represent second weekend off. Define $\bar{y} = y_1, \ldots, y_k$ to be the number of employees of a certain level of expertise that are assigned to pattern k. Create a one-column matrix $\bar{d}$, that represents the number of employees of a certain analyst type needed per day of the week such that the number of rows of d match the number of rows of matrix Y. The values of $\bar{d}$ is the value of $\{A, B, C\}$ for L1, L2, and L3 analyst respectively, which is obtained from the optimization algorithm. The values of $\bar{d}$ may or may not be all the same because the number of employees needed per day of the week could vary. The integer programming formulation to solve for the decision variables $y_1, \ldots, y_k$ is as follows

ALGORITHM 1: Genetic Algorithm for Optimization

Input: Number of sensors N, maximum number of analysts of type L1, L2 and L3 $max(A) = max(B) = max(C) = N$, total number of significant alerts generated M, upper bound on the risk % that the cybersecurity system can tolerate p, the lower and upper bound for the number of sensors that can be allocated to an analyst L_a and U_a, L_b and U_b , and L_c and U_c , the average time (in hours) taken by an analyst with L1, L2, and L3 level experience t_a, t_b, and t_c respectively, analyst utilization requirement (95%), and the lower and upper bound for the proportion of L1, L2, and L3 analysts in the organization L_A and U_A, L_B and U_B, and L_C and U_C respectively.

Output: The number of analysts of L1, L2, and L3 type A, B, C respectively, sensor-to-analyst allocation for L1, L2, and L3 type analyst $x_{a,n}$, $x_{b,n}$, and $x_{c,n}$ respectively, the number of significant alerts analyzed by analyst $m_{a,n}$, $m_{b,n}$, and $m_{c,n}$, and the total time spent by an analyst while engaged in alert investigation during a day (24 hr period) T_a, T_b, and T_c.

Step 0: Initialize a population of μ individuals (parents)
Step 1: Evaluate μ individuals (parents)
for *each* μ, **do**
    check number of sensors allocated to analyst constraints (Equation 6);
    check analyst proportion constraints (Equation 7);
    retain only feasible μ individuals;
end
Step 2: Evaluate the feasible μ individuals
$timeindex$ = 0;
repeat
    for *each feasible* μ **do**
        Generate and queue the alerts;
        Analyze alerts using a first-in-first-out rule;
        Calculate total time that an analyst was engaged using Equation 9;
        $timeindex$ ++ (1 hr time steps);
    end
until $timeindex$ = *24*;
for *each feasible* μ **do**
    if $Risk \leq p$ (Equation 8) and $analyst\ utilization \geq 95\%$ (Equation11);
    then
        Retain those μ individuals;
        Obtain number of analysts, Equation 3;
        Obatin objection function, Equation 2;
        Store the individual from μ that corresponds to the best objective function;
    end
end
Step 3: Continue the search for the optimal solution
repeat
    Generate λ offsprings from μ parents;
    Evaluate the λ offsprings using Step 1 and Step 2;
    Replace the population with μ individuals selected from the original parents and new offsprings;
    Continue the search;
until *Stopping Criteria*;
return *Best individual among all* μ *and* λ *(sensor-to-analyst allocation), and the optimal values of* $\{A, B, C\}$

$$Min \quad y_1 + y_2 +, \ldots, + y_k \tag{14}$$

Table II. Y matrix of feasible 12 hr schedules with every other weekend off

S	0	0	1	1	...
-	-	-	-	-	...
S	0	0	0	1	...
M	1	0	1	1	...
T	1	1	1	1	...
W	1	1	0	0	...
T	0	1	0	1	...
F	0	0	1	0	...
S	0	1	0	0	...
-	-	-	-	-	...
S	1	0	0	0	...
M	0	1	1	1	...
T	0	1	1	1	...
W	1	0	1	1	...
T	1	0	1	0	...
F	1	1	0	0	...
S	0	0	0	0	...
-	-	-	-	-	...
S	0	0	1	1	...

ALGORITHM 2: Cyclic Days-Off Staffing Algorithm

Input: Days-off pattern Y, and the number of analysts of L1, L2, and L3 type A, B, C respectively.
Output: For a given type of analysts from L1, L2, and L3, days-off schedule patterns and the number of personnel who follow that pattern, $\bar{y} = y_1, \ldots, y_k$.
Step 0: Select an analyst category to schedule and set values of $\bar{d}$ = A or B or C
Step 1: Solve a linear programing (LP) relaxation
for *each* $\bar{d}$, **do**
  Solve a LP relaxation of Equations 14 and 15;
  Obtain solution $y\acute{}_1, \ldots, y\acute{}_k$;
  if $y\acute{}_1, \ldots, y\acute{}_k$ = Integers **then**
    STOP;
  else
    add
    $y_1+, \ldots, +y_k = \lceil y\acute{}_1+, \ldots, +y\acute{}_k \rceil$;
    solve LP relaxation again;
  end
end
return $y_1, \ldots, y_k$ *and corresponding days-off patterns.*

subject to

$$\mathbf{Y}\bar{\mathbf{y}} \geq \bar{d} \tag{15}$$

$$\bar{y} \geq 0 \tag{16}$$

The above integer program is executed separately for each analyst type L1, L2, and L3. The integer programming formulation can be solved optimally. It should be noted that the number of possible patterns in Y is quite large that satisfy the constraints given above. Therefore, a days-off scheduling heuristic can also be used to solve the scheduling model [Pinedo 2009].

ALGORITHM 3: Simulation Algorithm to Calculate Risk and Analyst Utilization

Input: Number of sensors N, the number of analysts of L1, L2, and L3 type A, B, C respectively, sensor-to-analyst allocation for L1, L2, and L3 type analyst $x_{a,n}$, $x_{b,n}$, and $x_{c,n}$ respectively, total number of significant alerts generated M, the average time (in hours) taken by an analyst with L1, L2, and L3 level experience t_a, t_b, and t_c respectively.

Output: 95% confidence intervals on risk $p\%$, analyst utilization in % the number of significant alerts analyzed by analyst $m_{a,n}$, $m_{b,n}$, and $m_{c,n}$, and the total time spent by an analyst while engaged in alert investigation during a shift T_a, T_b, and T_c.

Step 1: Simulate a day's alert investigation by analysts
$timeindex$ = 0;
repeat
  for *a given sensor-to-analyst allocation* **do**
    Generate alerts using a probability distribution and queue them based on arrival time;
    Analyze alerts using a first-in-first-out rule;
    Calculate total time that an analyst was engaged using Equation 9;
    $timeindex$ ++ (1 hr time steps);
  end
until $timeindex$ = *24*;
Calculate Risk by using left side of Equation 8 and dividing by M, and substituting the result in % into the right side of Equation 1;
Calculate analyst utilization using left side of Equation11;
Step 2: Replicate the simulation to calculate confidence intervals on risk and analyst utilization
repeat
  Step 1 using newly generated alerts from the probability distribution;
  Continue calculating and storing the risk and analyst utilization for each simulation run;
until *the number of simulation replications have been completed*;
return *Confidence interval on risk and analyst utilization*

5.3. Simulation Module

The simulation module is used to validate the output of the optimization and the scheduler module. The simulation algorithm is presented under Algorithm 3. It is also used as a stand-alone module to perform sensitivity analysis. The inputs to the simulation module are as follows.

(1) Number of sensors, alert characteristics such as alert generation rate for the sensors,
(2) Analyst characteristics such as time taken to analyze an alert based on the experience,
(3) Number of analyst and sensor-to-analyst allocation for a given day, which are the outputs of the optimization model,
(4) The shift and days-off schedule for analysts, which are the outputs of the scheduling model,
(5) The assumptions of the optimization model that are also valid for the simulation model as well.

The outputs of the simulation model measure the utilization of each analyst per shift per day and the goodness of the cybersecurity analyst allocation model in terms of the risk. The alert generation is done using a probability distribution such as Poisson or uniform for the arrival of the alerts. The simulation module is run several times to obtain 95% confidence intervals on the overall utilization and the risk level that can be attained for a given sensor-to-analyst allocation that was determined by the optimization module.

5.3.1. Sensitivity Analysis. The simulation model can be used to perform stand-alone sensitivity analysis on the model parameters. For example, the alert generation rate, sensor-to-analyst allocation policy, number of sensors, and proportion of experience level mix among the analysts are varied to see the effects on both individual utilization of analysts and the overall risk level of the system. A design of experiment study is conducted to study the effects of varying the number of sensors, alert generation rate, and the number of personnel. Also, a baseline model (today's sensor-to-analyst allocation) executed via simulation can be compared to that of the optimization model in which both analyst utilization and the overall risk level can be measured. The next section presents the results from the simulation, optimization, and scheduling models along with sensitivity analysis.

6. RESULTS

The following section first presents the results of the stand-alone simulation studies that were performed to obtain several useful insights about the cybersecurity analyst scheduling problem. These insights were then used in the development of the optimization model, which was further validated using simulation. A sample days-off schedule is presented for the 10 sensor - 5 (L3 level) analyst optimization model results. A separate section provides the results from the design of experiments study that analyses the sensitivity of the modeling parameters. A summary of all the experiments is provided in Table III, which were conducted with both uniform and Poisson distribution for the average alert generation rate per sensor.

6.1. Results from Simulation Studies

The first stand-alone simulation study was conducted with 10 sensors. Two types of analyst experience-level mix were studied - 1) all employees are highly trained (L3 level) before they are assigned to the 10 sensor(s), and 2) the analyst group has a mix of 40% L1, 30% L2 and 30% L3 level analyst who work on the 10 sensors. Three types of sensor-to-analyst allocation strategies were studied: 1) one-on-one allocation, 2) multiple sensors to analysts (L1 analyses 1-2 sensors, L2 analyses 2-3 sensors and L3 analyses 4-5 sensors), and 3) all analysts were assigned all sensors. The final variation was in the total number of analysts in which two levels were studied - with 10 (10 L3 vs 4-L1, 3-L2, 3-L3) and 5 (5 L3 vs 2-L1, 1-L2, 2-L3) analysts respectively.

Tables IV-VIII provide the data for the above simulation studies with 10 analysts. The Tables IV- VI provide the sensor-to-analyst allocation for the three cases: Case 1) one-on-one allocation, Case 2) multiple sensors to analysts (L1 analyses 1-2 sensors, L2 analyses 2-3 sensors and L3 analyses 4-5 sensors), and Case 3) all analysts were assigned all sensors. Tables VII-VIII provide the time taken in hours by the analyst to investigate an alert from a given sensor based on their training. Table VII provides time to analyze data of 10 highly trained analysts (L3 level), however, they are trained only on one specific sensor because training all analysts on all sensors is impractical to implement. Table VIII provides time to analyze data of 40% L1, 30% L2 and 30% L3 level analysts who work on the 10 sensors. Note that the L3 level analysts take less time than L2, and L2 take less time than L1. This data was obtained based on the information on analyst workload given under the fixed parameters section of the paper. A similar set of tables were created for the five analyst scenario.

The significant alert generation rate was 1% of the entire alerts generated. Each sensor is said to generate about 15000 alerts per day and 1% (approx. 150) are deemed to be significant, which must be analyzed by a cybersecurity analyst. The remaining alerts are considered insignificant. Therefore, the average significant alert generation per hour was taken to be 6.5. A uniform distribution (U(0,13)) and a Poisson distribu-

Table III. Summary of Experiments.

Experiment	Sensor to analyst allocation	Expertise mix	Number of sensors	Analyst to sensor ratio	Results
Simulation					
1	Case 1: one-one-one (Table IV)	All L3	10	1	Table IX
2		40% L1, 30% L2, 30% L3	10	1	Table IX
3	Case 2: multiple sensors to analysts (Table V)	All L3	10	1	Table IX
4		40% L1, 30% L2, 30% L3	10	1	Table IX
5	Case 3: all sensors to all analysts (Table VI)	All L3	10	1	Table IX
6		40% L1, 30% L2, 30% L3	10	1	Table IX
7	Case 1: one-one-one (Table IV)	Not applicable	10	0.5	Table X
8		Not applicable	10	0.5	Table X
9	Case 2: multiple sensors to analysts (Table V)	All L3	10	0.5	Table X
10		40% L1, 30% L2, 30% L3	10	0.5	Table X
11	Case 3: all sensors to all analysts (Table VI)	All L3	10	0.5	Table X
12		40% L1, 30% L2, 30% L3	10	0.5	Table X
Design of Experiment					
13 - 35	Case 3: all sensors to all analysts	40% L1, 30% L2, 30% L3	10, 25, 50, 75, 100	0.5, 0.6, 0.75, 0.9,1	Table XI and Table XII
Optimization					
Experiment	Sensor to analyst allocation	Expertise mix	Number of sensors	Risk	Results
36 -38	Optimization output	All L3	10	5%, 25%, and 45%	Table XIII and Table XIV
39 - 41	Optimization output	20-40% L1, 30-50% L2, and 30-40% L3	10	5%, 25%, and 45%	Table XV and Table XVI
42 - 49	Optimization output	All L3	25, 50, 75, 100	5%, 25%	Table XVII and Table XVIII
50 - 57	Optimization output	20-40% L1, 30-50% L2, and 30-40% L3	25, 50, 75, 100	5%, 25%	Table XVII and Table XVIII
Validation					
58 - 61	Optimization output	Optimization output	25, 50, 75, 100	Risk is an output	Table XIX
Scheduling					
62	Optimization output	All L3	10	Number of Analysts = 10	Table XX

Table IV. One-on-one sensor-to-analyst allocation

Sensor →	1	2	3	4	5	6	7	8	9	10
Analyst ↓										
1	1	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0
7	0	0	0	0	0	0	1	0	0	0
8	0	0	0	0	0	0	0	1	0	0
9	0	0	0	0	0	0	0	0	1	0
10	0	0	0	0	0	0	0	0	0	1

Table V. Multiple sensor-to-analyst allocation

Sensor →	1	2	3	4	5	6	7	8	9	10
Analyst ↓										
1	1	0	0	1	0	0	0	0	0	0
2	0	1	0	0	0	0	1	0	0	0
3	0	0	1	0	1	0	0	0	0	1
4	0	0	0	1	0	0	0	0	1	0
5	0	1	0	0	1	0	0	0	0	0
6	0	0	0	0	0	1	0	1	0	0
7	0	0	1	0	0	0	1	0	0	0
8	0	0	0	1	0	0	0	1	0	0
9	0	0	0	0	0	1	0	0	1	0
10	1	0	0	0	1	0	0	0	0	1

Table VI. All analysts can analyze all sensors

Sensor →	1	2	3	4	5	6	7	8	9	10
Analyst ↓										
1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1

Table VII. Time taken in hrs to analyze an alert. Analysts are highly trained on one sensor

Sensor →	1	2	3	4	5	6	7	8	9	10
Analyst ↓										
1	0.1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
2	0.9	0.1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
3	0.9	0.9	0.1	0.9	0.9	0.9	0.9	0.9	0.9	0.9
4	0.9	0.9	0.9	0.1	0.9	0.9	0.9	0.9	0.9	0.9
5	0.9	0.9	0.9	0.9	0.1	0.9	0.9	0.9	0.9	0.9
6	0.9	0.9	0.9	0.9	0.9	0.1	0.9	0.9	0.9	0.9
7	0.9	0.9	0.9	0.9	0.9	0.9	0.1	0.9	0.9	0.9
8	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.1	0.9	0.9
9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.1	0.9
10	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.1

Table VIII. Time taken in hrs to analyze an alert. Analysts with a mixed level of expertise

Sensor →	1	2	3	4	5	6	7	8	9	10
Analyst ↓										
L3	0.1	0.09	0.08	0.11	0.1	0.09	0.08	0.07	0.09	0.1
L3	0.1	0.08	0.11	0.11	0.1	0.09	0.09	0.08	0.09	0.1
L3	0.09	0.09	0.08	0.08	0.1	0.08	0.08	0.07	0.1	0.1
L2	0.13	0.14	0.15	0.13	0.12	0.14	0.15	0.12	0.12	0.13
L2	0.14	0.14	0.14	0.13	0.14	0.14	0.15	0.13	0.14	0.13
L2	0.13	0.14	0.15	0.13	0.12	0.12	0.15	0.12	0.12	0.13
L1	0.2	0.21	0.23	0.19	0.18	0.2	0.21	0.19	0.18	0.19
L1	0.2	0.21	0.23	0.2	0.18	0.2	0.22	0.2	0.19	0.19
L1	0.19	0.21	0.23	0.19	0.18	0.19	0.21	0.19	0.18	0.19
L1	0.2	0.2	0.23	0.2	0.18	0.2	0.21	0.19	0.18	0.19

Table IX. Risk and analyst utilization for 10 sensors and 10 analysts

Risk in %	Case 1	Case 2	Case 3
10-L3 level analysts only	2.2	60.9	82.4
4-L1, 3-L2, 3-L3	15.3	2.4	2.2
Analyst Utilization in %			
10-L3 level analysts only	76.1	102.7	103.0
4-L1, 3-L2, 3-L3	89.5	98.4	99.4

Case 1: One-on-one sensor assignment.
Case 2: Multiple sensors to analyst.
Case 3: All analysts can work on all sensors.

tion (P(6.5)), was used to generate the significant alerts per hour for this study. The results below are presented for the uniform distribution.

Table IX presents the results of the first simulation study with 10 sensors and 10 analysts with varying proportions of experience and allocation strategy. The average values of overall risk and analyst utilization are presented from 50 simulation runs. First, the observations when all 10 analysts are at L3 level are presented. It is observed from Table IX that the one-on-one analyst-to-sensor allocation (Case 1) with all L3 level analysts is ideal to achieve the lowest risk of 2.2 %, however, the average analyst utilization is about 76.1%. This allocation strategy is certainly inefficient because the analysts are not being fully utilized, but more importantly, it is impractical to have as many analysts as the number of sensors. In Case 3, the risk level is very high (82.4%) when analysts are allowed to work on all sensors, which is also evident in Table VII where analysts take more time to analyze alerts from sensors that they are not trained on. It is also impossible to train all analysts on all sensors. In Case 2, highly trained analysts work on multiple sensors but they are trained on only one of them. The strategy helps to increase their utilization but the risk level was high at 60.9%. The utilization was very high with numbers around 100%. Some of the utilization numbers are above 100%. This is because the analyst would finish the alert investigation that is on-going even if their shift time has ended. So they tend to stay a few minutes longer. Hence, the ratio of the time the analyst was utilized to the total length of their shift time could exceed 1. At the moment it is assumed that analysts immediately pick up the next alert in the queue as and when an on-going alert has been investigated. If no alert is available then the analyst would be idle. Personal break times are not modeled although a few breaks can be easily introduced in the simulation by adding scheduled analyst down-time.

Second, results from Table IX when there is a mix of analysts experience such as 4-L1, 3-L2, 3-L3 are presented. It can be observed that the one-on-one allocation (Case 1) is a poor strategy because allocating a dedicated sensor to a junior analyst will result is many unanalyzed alerts due to the slow pace of work. This resulted in a risk of 15.3%. Having everyone work on all sensors (Case 3) was found to be the best strategy but it is impractical to train everyone on every sensor. Hence, the strategy that is viable and makes the most sense is to have a mix of analyst experience (Case 2) and train them on a set of multiple sensors by following the rule in which L1 analyses 1-2 sensors, L2 analyses 2-3 sensors and L3 analyses 4-5 sensors. The average risk level is 2.4% and the overall average analyst utilization is about 98.4%.

It is observed that Case 2 in Table IX still has equal number of analysts and sensors, which is a very expensive strategy, hence, not very realistic to implement. In the following simulation study, all the parameters of the above simulation were held constant with the only change in the number of analysts from 10 to 5. Table X presents the results of the second simulation study with 10 sensors and 5 analysts. The one-on-one allocation strategy is not applicable for this study. The ideal strategy, given only

Table X. Risk and analyst utilization for 10 sensors and 5 analysts

Risk in %	Case 1	Case 2	Case 3
5-L3 level analysts only	NA	30.9	30.6
2-L1, 1-L2, 2-L3	NA	46.2	48.4
Analyst Utilization in %			
5-L3 level analysts only	NA	99.6	99.7
2-L1, 1-L2, 2-L3	NA	99.7	100.0

Case 1: One-on-one sensor assignment.
Case 2: Multiple sensors to analyst.
Case 3: All analysts can work on all sensors.
NA: Not Applicable.

5 analysts, was to have all five L3 analysts (senior level), which yielded an average risk of 30%. Since the queue length of the alerts were long due to fewer analysts, there was not a statistically significant difference between allocating multiple sensors (4-5 sensors to L3 level analysts) as in Case 2 and all sensors to all L3 analysts as in Case 3. The utilization in both cases was found to be almost 100%.

Since an organization tends to have a mix of analysts with various levels of expertise, a mix of 2-L1, 1-L2, 2-L3 level analysts was chosen for the 10 sensor and 5 analyst simulation study. Since L1 and L2 tend to work slower than L3, the risk increased from an average of 30% with all L3 level analysts to about an average of 47% with the analyst mix for both Case 2 and Case 3. There was no statistically significant difference between Case 2 and Case 3 in mitigating risk and in analyst utilization. The only way to minimize risk in a situation with a mix of analysts with different experience would be to increase the number of personnel. For a given risk, one of the ways to study the tradeoff between having a few L3 level analysts and a larger group of analysts with a mix of experience is to study the cost of hiring, which is one of the future works of this research. The utilization in both cases was found to be almost 100%.

6.2. Design of Experiments

The above simulation studies provided the basis for conducting further simulations of cybersecurity analyst scheduling scenarios. A design of experiment (DOE) test was conducted with the alert generation rate fixed at (U(0,13)) to study the impact of 1) increasing the number of sensors, and 2) varying the analyst/sensor ratio on the risk measure of the system. The analyst mix was maintained at 40% L1, 30% L2 and 30% L3 levels. Since the number of sensors is scaled up, all analysts were allowed to work on all sensors because there are several possible combinations of sensor-to-analyst allocation if one were to choose only a few sensors for each analyst. The main assumption of this study is that all analysts are also trained on all sensors. The time taken by analysts to investigate the alerts is the same as the previous studies, and the details on the time to investigate are the same as those provided under the fixed parameters section of this paper. The DOE treatment combinations are as follows.

(1) Vary the number of sensors. The DOE treatment levels are 10, 25, 50, 75, and 100.
(2) Vary the analyst/sensor ratio. The DOE treatment levels are 0.5, 0.6, 0.75, 0.9, and 1.

Both risk and analyst utilization were observed in all the simulations and the average values of 50 simulations per treatment combination is presented in Tables XI and XII respectively. It can be observed from Table XI that as the number of sensors is increased there is no significant change in the Risk %. This is also shown in Figure 4. This can be explained as follows. With the increase in the number of sensors and proportionally the number of analysts, both the arrival rate of alerts and the service

Table XI. Risk in %

Number of sensors →	10	25	50	75	100
analyst/sensor ratio ↓					
0.5	48.4	46.6	47.5	46.8	48.1
0.6	25.4	26.2	26.4	25.8	26.4
0.75	10.8	10.5	11.2	10.9	10.3
0.9	4.5	4.1	4.2	4.1	3.9
1	2.2	2.2	2.8	2.6	2.5

Table XII. Analyst Utilization in %

Number of sensors →	10	25	50	75	100
analyst/sensor ratio ↓					
0.5	100	99.6	98.9	99.4	99.9
0.6	99.8	98.5	99.5	99.5	100
0.75	99.5	99.2	99.5	98.8	100
0.9	100	99.1	100.2	99.6	99.9
1	99.4	100	100	99.5	100

rate of analysts are proportionally increased. If we assume that both arrival rate of alert from a sensor and service rate of an analyst to be Poisson distributed then the sum of several Poisson distributions is also Poisson distributed. From queueing theory, for a M/M/1 queue with Markovian arrival rate (total arrival rate of the system of several sensors), Markovian service rate (total service rate of the system with several analysts), and 1 service personnel (represents the combined service of all service personnel), the queue length and waiting time in the queue are dependent on the arrival and service rates [Gross et al. 2008]. The ratio of the arrival to service rate ρ remains the same when the number of sensors and analysts are increased proportionally. Hence, the risk % will remain the same. The only way to reduce the risk is to increase the service rate by holding the arrival rate fixed (increase the number of analysts). The average analyst utilization over 50 simulations in the above DOE study was found to be almost 100% as shown in Table XII. Also, in the above experiment, all analysts were trained on all sensors, which is neither cost-effective nor is practical to implement.

In summary, both from the above DOE study and the simulations done earlier, it is clear that a mix of analysts is required for an organization of cybersecurity analysts, and realistically only a few sensors can be allocated to an analyst based on their level of expertise. Hence, it is imperative to have a model that can assign sensors to analysts such that minimum number of analyst with the right experience mix are hired for a given upper bound on risk%, number of sensors, and alert generation rate. In the next section, the paper presents the results of the optimization model that can determine an efficient sensor-to-analyst allocation strategy to meet the above requirements.

In summary, the simulation studies indicated the following:

(1) One-on-one allocation of sensor to analyst is the best strategy for minimizing risk, however, the strategy is impractical to implement.
(2) Do not allocate a dedicated sensor to a junior analyst. Junior analysts are slower in investigating alerts and many significant alerts will remain unanalyzed.
(3) When there are fewer analysts in comparison to the number of sensors (analyst to sensor ratio = 0.5), having all L3 level senior analysts is the best strategy than having a mix of experience among the analysts.
(4) A single queue for alerts waiting for investigation is used. When multiple sensors are assigned to an analyst and multiple analysts are assigned to sensors, the sensors are grouped and allocated to the analyst group. A single queue of alerts is

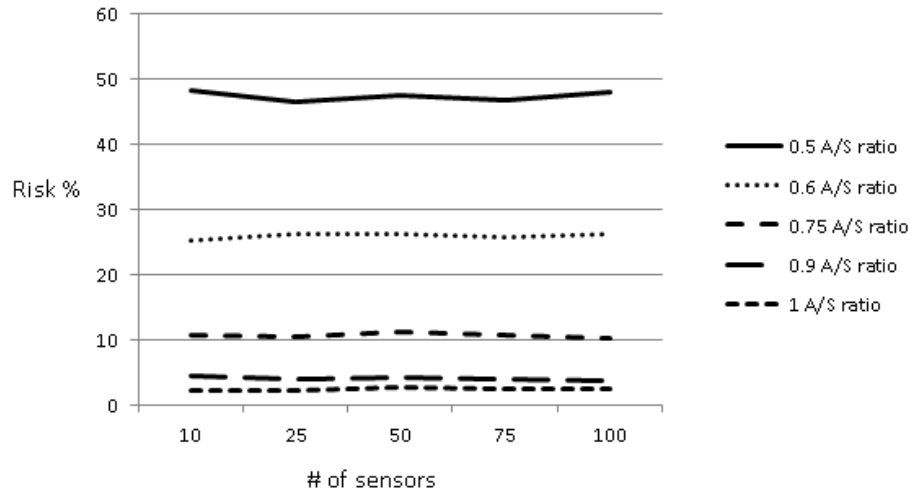

Fig. 4. Risk % vs number of sensors for varying analyst/sensor ratios.

formed within this group based on the time of arrival. Alerts are drawn from the queue on a first-in-first-out basis.

(5) An organization that aims to have a mix of personnel with L1, L2 and L3 level experience should aim to find the right mix of expertise for a given upper bound on risk level that it wishes to maintain.

The last point in the above summary serves as a motivation for developing the optimization model, which aims to identify the right mix of expertise and a sensor-to-analyst allocation strategy for a given upper bound on risk level.

6.3. Results from Optimization Model

The optimization model presented earlier as a mixed-integer programming algorithm was run in two ways- 1) all L3 level analysts, and 2) by specifying a range for the proportion of L1, L2, and L3 level expertise in the organization (20-40% L1, 30-50% L2, and 30-40% L3 level analysts). Again, 10 sensors were used in this study with the same rate of alert generation, and the assumptions that were used in the simulation model were applied to the optimization model. The optimization model was solved using genetic algorithm heuristics. Three risk levels were studied- 5%, 25%, and 45% and the best sensor-to-analyst allocation was derived for each of them.

For the first optimization model (no expertise mix), the algorithm was initiated with 15 L3 level analysts only. For the second optimization model (with expertise mix), the algorithm was initiated with 15 analysts in all, and a random allocation into each of the L1, L2, and L3 level was made that met the constraints from the upper and lower bound on the analyst proportions (20-40% L1, 30-50% L2, and 30-40% L3 level analysts). At each iteration of the algorithm, the total number of analysts would be changed within a range of (5-15) and a random allocation into L1, L2, and L3 level was made that met the above proportionality constraint.

The value of the number of initial populations of individual solutions μ was set to be 10 for the genetic algorithm heuristics. This was followed by a random sensor-to-analyst allocation that followed the rules on how many sensors can be allocated to a certain level of analyst expertise. The next step was to generate alerts and allocate

Table XIII. Results of optimization model without experience mix. All L3 level analysts

Risk in %	5%	25%	45%
All L3 analysts	7	5	3
Utilization of analyst	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$
Number of sensors	10	10	10
Alert generation rate	U(0,13)	U(0,13)	U(0,13)
Number of sensors per L3 analyst	4-5	4-5	4-5

Table XIV. Sensor-to-analyst allocation, 5% risk and all L3 level analysts

Sensor $\rightarrow$	1	2	3	4	5	6	7	8	9	10
Analyst $\downarrow$										
1	1	1	0	0	0	0	0	0	1	1
2	0	0	0	0	0	0	1	1	1	1
3	1	1	1	1	1	0	0	0	0	0
4	0	0	0	0	0	1	1	1	1	1
5	0	0	1	1	1	1	0	0	0	0
6	0	0	0	0	1	1	1	1	0	0
7	1	1	1	1	0	0	0	0	0	0

them to analysts, and the process was simulated for a day's operation as given in Algorithm 1. Crossover and mutations, which are specific steps of genetic algorithm, were used in the generation of offsprings during the execution of the optimization algorithm. The genetic algorithm was run until no significant improvement in the result (risk % and analyst utilization %) was observed for 7 consecutive iterations, which was the stopping criterion used in this paper. The best solution from optimization for all the risk levels of 5%, 25%, and 45% is presented in Table XIII for no expertise mix scenario (all L3 level) and in Table XV for the expertise mix scenario, which are explained below.

Table XIII provides the results of the optimization model in which the proportion of analysts with different expertise was not specified (all L3 level analysts). The table provides the details on the number of L3 level analysts that are needed to achieve a given upper bound on risk. The upper bound p on risk was varied in 3 levels, 5%, 25%, and 45%, as given in the table, and 50 one-day (24-hour) runs of the optimization model were executed. It should be noted that each run of the optimization algorithm constitutes several iterations of the genetic algorithm. The number of sensors allocated per analyst was in the order of 4-5, which was in line with the constraints on L_c and U_c numbers that were specified for L3 level analysts. Table XIV presents a sample of the sensor-to-analyst allocation for a risk upper bound of 5% where all analysts had L3 level expertise. It can be observed from the table that each analyst had 4-5 sensors allocated to them as per the above constraint. Utilization of analysts was above 95% in each of the optimization runs, which is also one of the constraints to be met by the optimization model. Also, it can be noticed that each sensor was allocated to three L3 level analysts.

Table XV provides the results of the optimization model in which the proportion of analysts with different expertise was specified (20-40% L1, 30-50% L2, and 30-40% L3 level analysts). This is a more practical case in the real-world wherein an organization would prefer a mix of expertise in order to provide promotion opportunities and the training of junior analysts by senior analysts. The obvious implication of having a mix of expertise is that for the same level of risk there is an increase in the total number of personnel as compared to the previous case in which all was L3 level analysts. For example, for a risk of 25% either 5- L3 level analysts (see Table XIII) or a total of 6 (1-L1, 3-L2, 2-L3) personnel (see Table XV) is needed. The lower and upper bounds

Table XV. Results of optimization model specifying analyst experience mix

Risk in %	5%	25%	45%
Analysts experience mix	2-L1, 3-L2, 3-L3	1-L1, 3-L2, 2-L3	1-L1, 2-L2, 2-L3
Total number of analysts	8	6	5
Utilization of analyst	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$
Number of sensors	10	10	10
Alert generation rate	U(0,13)	U(0,13)	U(0,13)
Number of sensors per L1 analyst	1-2	1-2	1-2
Number of sensors per L2 analyst	2-3	2-3	2-3
Number of sensors per L3 analyst	4-5	4-5	4-5

Table XVI. Sensor-to-analyst allocation, 5% risk and 2-L1, 3-L2, and 3-L3 level analysts

Sensor $\rightarrow$	1	2	3	4	5	6	7	8	9	10
Analyst $\downarrow$										
L3	1	1	0	0	0	1	0	0	1	1
L3	0	0	0	0	0	1	1	1	1	1
L3	1	1	1	1	1	0	0	0	0	0
L2	0	0	1	1	1	0	0	0	0	0
L2	0	0	0	0	0	0	1	1	1	0
L2	0	0	0	1	1	0	1	0	0	0
L1	0	0	1	0	0	0	0	0	0	0
L1	0	0	0	0	0	1	0	0	0	0

on the number of sensors allocated to each type of expertise level is also provided in Table XV. Table XVI presents a sample of the sensor-to-analyst allocation for a risk upper bound of 5% where analyst expertise proportion was 2-L1, 3-L2, and 3-L3. It can be observed from the table that each L3 analyst had 4-5 sensors allocated to them, each L2 analyst had 2-3 sensors allocated to them, and each L1 analyst had 1-2 sensors allocated to them. These conform to the lower and upper bounds set in the optimization model for the number sensors that can be allocated to each analyst based on their level of expertise. Utilization of analysts was above 95% in each of the optimization runs, which is one of the constraints of the model.

6.4. Scalability of Optimization

The optimization model was run for different number of sensors to study its scalability. The code was written in Matlab. The computational time increases as the number of sensors increases. For example, for 100 sensors, the genetic algorithm took just under 3 hrs to find a good enough solution that satisfied the stopping criteria on an 8GB RAM processor. The stopping criteria used in genetic algorithm was to stop the search for solution when 7 consecutive iterations of the algorithm did not improve the previously found best solution. It should be noted that it is not possible to prove optimality in heuristics. Both optimization models 1) all L3 analysts, and 2) mix of expertise following (20-40% L1, 30-50% L2, and 30-40% L3) were executed for 50 runs for 10, 25, 50, 75, and 100 sensors. The results are presented in Tables XVII and XVIII for 5% and 25% risk respectively. In each of the optimization runs, the sensor-to-analyst allocation matrix that corresponds to the best solution from the genetic algorithm was stored as the final output of the optimization algorithm.

The results in Tables XVII and XVIII match those in Table XI. For example, at 25% risk in Table XI the analyst/sensor (A/S) ratio was 0.6 for the simulation with 40% L1, 30% L2 and 30% L3 analysts. The same result is seen with optimization in Table XVIII where the A/S ratio for the analyst with experience mix is about 0.6 for any number of sensors. The expertise mix for optimization was constrained to the following 20-40% L1, 30-50% L2, and 30-40% L3. It can also be observed from Tables XVII and XVIII that the A/S ratio with all L3 analysts is about 0.7 (5% risk) and 0.5 (25% risk) respectively.

Table XVII. Number of analysts needed at 5% risk under varying number of sensors

Risk in %	5	5	5	5	5
Number of sensors	10	25	50	75	100
Number of all L3 analysts	7	17	33	52	67
A/S ratio- all L3 level analysts	0.7	0.68	0.66	0.69	0.67
Analysts with experience mix L1	2	5	10	15	20
Analysts with experience mix L2	3	7	15	23	30
Analysts with experience mix L3	3	8	16	24	30
Analysts with experience mix- total	8	20	41	62	80
A/S ratio- all L3 level analysts	0.8	0.8	0.82	0.83	0.8
Utilization of analyst	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$
Alert generation rate	U(0,13)	U(0,13)	U(0,13)	U(0,13)	U(0,13)
Number of sensors per L3 analyst	4-5	4-5	4-5	4-5	4-5

Table XVIII. Number of analysts needed at 25% risk under varying number of sensors

Risk in %	25	25	25	25	25
Number of sensors	10	25	50	75	100
Number of all L3 analysts	5	13	25	38	48
A/S ratio- all L3 level analysts	0.5	0.52	0.5	0.51	0.48
Analysts with experience mix L1	1	3	6	8	10
Analysts with experience mix L2	3	6	12	20	30
Analysts with experience mix L3	2	6	15	20	20
Analysts with experience mix- total	6	15	33	48	60
A/S ratio- analysts with experience mix	0.6	0.6	0.66	0.64	0.6
Utilization of analyst	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$	$\geq 95\%$
Alert generation rate	U(0,13)	U(0,13)	U(0,13)	U(0,13)	U(0,13)
Number of sensors per L1 analyst	1-2	1-2	1-2	1-2	1-2
Number of sensors per L2 analyst	2-3	2-3	2-3	2-3	2-3
Number of sensors per L3 analyst	4-5	4-5	4-5	4-5	4-5

Clearly, more analysts are needed when there is a mix of expertise in comparison to having all L3 senior level analysts only, for the same amount of risk.

6.5. Validation of Optimization using Simulation

The sensor-to-analyst allocation results that were obtained from the optimization model (see Tables XVII and XVIII for 5% and 25% risk respectively) were run 50 times each using the simulation model, and the 95% confidence intervals on risk and analyst utilization are presented in Table XIX. The simulation followed Algorithm 3. For each simulation, the number of sensors, and the number of analysts at L1, L2, and L3 levels were fixed from Tables XVII and XVIII for 5% and 25% risk respectively. Alerts were generated using both U(0,13) and Poisson(6.5) distributions, and results from the uniform distribution are presented here. The alert allocation and investigation continued for a 24 hr period and the 95% confidence intervals on risk and analyst utilization were determined. Several simulation scenarios were studied and the results from a sample of four cases is presented in Table XIX. It can be observed that the average values of the risk and analyst utilizations match those that were set for the optimization algorithm. Thus, the optimization results were validated using the simulation model.

Figure 5 summarizes the plot between risk and analyst/sensor (A/S) ratio. It should be noted from Figure 4 that for a given A/S ratio, there is no significant change in risk as the number sensors increases. This means that as long as the rate of arrival of alerts and the rate of service by analysts remain the same regardless of the number of sensors, the queue length and the time that an alert awaits investigation will remain the same as proved by the queueing theory model for M/M/1 queues [Gross et al. 2008]. Figure 5 is an important chart that explains the relationship between risk and number of personnel to hire per day for a given number of sensors, which is expressed as

Table XIX. Confidence Interval on risk and analyst utilization for four sample sensor-to-analyst allocations.

Number of sensors	50	75	25	100
Analysts with experience mix L1	10	15	3	10
Analysts with experience mix L2	15	23	6	30
Analysts with experience mix L3	16	24	6	20
Average Risk in %	5.8	6.2	26.8	26.2
95 % Confidence Internal on Risk in %	4.9-6.7	5.1-7.3	25.2-28.4	24.8-27.6
Analyst Utilization in %	99.9	99.5	99.6	99.4
95 % Confidence Internal on Utilization in %	99.8-100	99.3-99.7	99.5-99.7	99.3-99.5

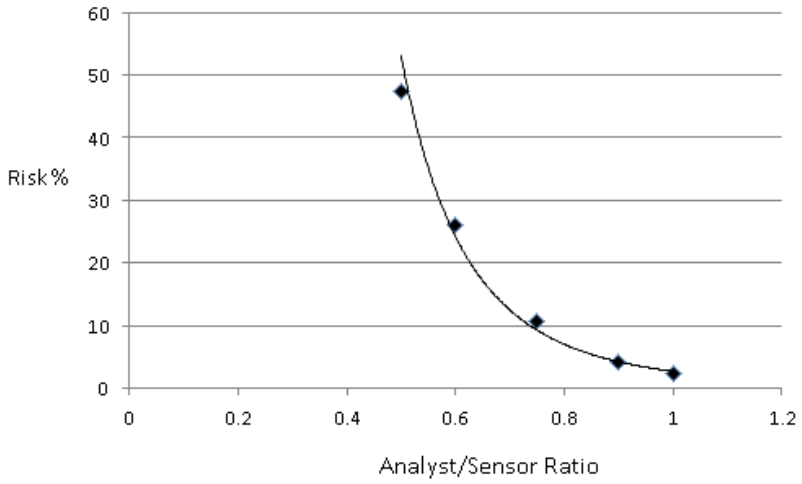

Fig. 5. Risk % vs Analyst/Sensor ratio.

A/S ratio. It should be noted that the plot holds good only for a given analyst mix that the organization desires, fixed average alert generation rate per sensor, fixed value of number of sensors that can be allocated to an analyst, fixed analyst mix in a work shift which depends on scheduling constraints, and fixed value of the service time to investigate an alert which depends on the analyst experience level. Clearly, risk depends on several factors. Hence, Figure 5 is a simplified plot of risk vs. A/S ratio in which the ratio of analysts to sensors determines the risk level when all other factors given above are kept constant. Therefore, it should not be construed that the optimization model can be simplified merely by adjusting the A/S ratio to build a certain capacity that can maintain a certain upper-bound on risk. The figure shows a non-linear relationship between risk and analyst/sensor ratio. As the A/S ratio drops, the risk increases dramatically. The above figure can guide the personnel hiring decision making process of an organization provided the other factors given above are maintained fixed at a pre-determined level. The total cost of personnel can be easily derived using personnel pay-scale, once the number personnel and their expertise level is known.

The computational complexity of the combined optimization and validation using simulation for determining the optimal sensor to analyst allocation for a given set of constraints is as follows. On a 16 GB RAM with a 3.2GHz processor, the genetic algorithm took about 0.1 hour for finding the best feasible solution for 10 sensors and close to 4 hours for 100 sensors. The population size was maintained at 20 for all iterations and the genetic algorithm was stopped when 7 successive iterations did not

yield a solution better than the best available solution at the time of termination. The best solution not only had the minimum number of analysts but also had the lowest risk that was below the upper bound set for risk in the optimization model.

In summary, the optimization model recommendations are as follows.

(1) The optimization model attempts to minimize the number of analysts. Therefore, it attempts to pick the highest number of L3 analysts and assigns them to the maximum number of sensors as allowed by the upper bound for L3 level analysts. However, in some instances this might create an infeasible solution for the L2 and L1 analysts, which may not fall within the given range for their proportions within an organization. In such instances, the L3 level is adjusted to find the best possible expertise mix that minimizes the total number of analysts and meets the constraints of the optimization problem.
(2) Since the optimization model is solved using heuristics, it is important to run several replications of the optimization model and choose the best solution and the corresponding sensor-to-analyst allocation. This is because heuristics do not guarantee an optimal solution. Even if an optimal solution was found there is no way to prove its optimality. Often times, one would settle for the good enough (or best available) solution at the time of stopping the algorithm.
(3) Junior L1 level analysts are never assigned a dedicated sensor. They are always paired with a group that has at least 1 senior level L3 analyst. This can be verified from table XVI.
(4) In general, having more L3 level analysts is better. However, one should be cautious about the cost of many L3 level analysts due to their high pay-scale, which was not explicitly modeled in this paper. A mix of analyst expertise could be more cost-effective (L1 pay-scale$<$L2 pay-scale$<$L3 pay-scale) even though a larger group will be needed in comparison to all L3 level analysts for the same level of risk.

6.6. Scheduling of Analysts

Days-off scheduling for cybersecurity analysts was run separately with inputs from the optimization model on the sensor-to-analyst allocation and the number of L1, L2, and L3 type analysts. Since the optimization model provides the number of analysts that are required to be working per day, the real-world system will have a few additional analysts in each of the L1, L2, and L3 level expertise to account for those who are on leave in any given day. As an example, let each analyst get at least 2 days off in a week and every other weekend off. An analyst works no more than 5 consecutive days. Also, analyst can work 12 hr shifts and complete 80 hrs between Sunday and Saturday over a 14-consecutive day period with $12 * 6 + 1 * 8 = 80$ hours (7 days of work in every 14 days starting on a Sunday and ending on a Saturday).

The minimum number of employees needed W as per the above constraints is given as follows.

$$W_1 \geq \lceil \frac{k_2 max(n_1, n_7)}{k_2 - k_1} \rceil \tag{17}$$

$$W_2 \geq \lceil \frac{1}{5} \sum_{j=1}^{7} n_j \rceil \tag{18}$$

$$W_3 \geq max(n_1, \ldots, n_7) \tag{19}$$

$$W = max(W_1, W_2, W_3) \tag{20}$$

Table XX. Days-off schedule for 10 L3 analysts

Day →	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S
1	X	X	X	X			X			X	X				X	X	X	X			X	
2	X	X		X	X	X					X	X			X	X		X	X	X		
3	X	X			X	X					X	X	X		X	X			X	X		
4	X	X				X	X			X			X	X	X	X				X	X	
5	X	X	X				X			X		X		X	X	X	X				X	
6			X	X	X			X	X				X	X			X	X	X			X
7				X	X			X	X	X	X			X				X	X			X
8			X		X	X		X	X		X	X					X		X	X		X
9				X		X	X	X	X			X	X					X		X	X	X
10			X				X	X	X	X			X	X			X				X	X

where k_1 weeks are off in k_2 weeks, and $n_1, \ldots, n_7$ is the number of employees needed on $Sunday, \ldots, Saturday$ respectively.

For a sample scenario of 10 sensors and 5 (all L3) analysts required per day, $k_1 = 1$, and $k_2 = 2$, and $n_1, \ldots, n_7 = 5$. The value of W is 10, which is the number of L3 employees that the organization must hire to meet the days-off constraints given above. Algorithm 2 was executed for several scenarios of sensor-to analyst allocation, and the results for the above sample scenario are given in Table XX. The algorithm identified 10 different days-off scheduling patterns, one for each of the 10 hired employees. The constraints given above can be easily verified in Table XX.

Shift scheduling was not performed and analysts were assumed to have non-overlapping shifts. Also, average alert generation rates throughout day was fixed (Poisson or uniformly distributed). For example, in two 12-hr non-overlapping shifts per day, $2X$ analysts are needed to cover the entire day, where X is the number of analysts determined per day by the optimization algorithm. However, alert generation rate can change from time-to-time. Consequently, more analysts are needed at certain hours of the day. In such a case, over-lapping shifts with different shift lengths can provide the required number of analysts to meet the hourly demand using shift-scheduling algorithms [Pinedo 2009]. The inputs to the above algorithm are 1) the number of available shift types and their time length, and 2) the demand per hour for the number of analysts required based on the alert generation rate and the time taken to analyze the alerts.

7. CONCLUSION

The paper presents an optimization model for cybersecurity analysts scheduling. The problem is modeled as a mixed-integer programming model with the objective to minimize the number of analysts and provide a sensor-to-analyst allocation, which minimizes risk and maintains risk under a given upper bound, achieves a given expertise mix, and obtains 95% utilization of analysts as desired by an organization. The model is very generic and can be used with any number of sensors, various alert generation rates, and different analyst characteristics such as their expertise level, which in turn affects the time taken to investigate an alert. Several meta-principles have been highlighted in the summary section of both the simulation and optimization model which serve as guidelines for setting up an effective cybersecurity organization with analysts investigating the most significant alerts that are generated from the sensor data. Risk as defined in this paper can be further mitigated with an increase in personnel hires. It was determined that risk level varied with analyst/sensor ratio in a non-linear way, and for a given A/S ratio, risk was independent of the number of sensors.

There are several future extensions to the scheduling methods presented in this paper. Cost was not explicitly modeled in this paper. If budget constraints are introduced then the optimization model could float the risk variable and determine the minimum

risk and the sensor-to-analyst allocation that can be achieved for a given number of analysts that can be hired within the budget. On the other hand, if a certain upper bound on risk is pre-set in addition to a budget constraint then either the optimization model may fail to find a feasible sensor-to-analyst allocation or several feasible sensor-to-analyst allocation solutions may be found and the one with the minimum cost and minimum risk can be chosen.

The sensor-to-analyst allocation model in this paper is static and does not adapt to changes in alert generation rates or changes to the expertise mix in a shift due to unforeseen reasons such as absenteeism of analysts or excessive alert generation from an intrusion. One of the future extensions to the model will include a dynamic workforce that will be available on-call basis. The dynamic workforce will augment the static scheduled workforce in every shift as and when needed. This will require a control theoretic view for dynamic modeling of the sensor-to-analyst allocation, which adapts to changes in the model parameters. An all proactive dynamic adaptive workforce model will be another future research, in which workforce is scheduled based on current level of alerts and predicted future workload for alert investigation. A powerful alert prediction model for determining the future workforce needs is required for such a dynamic model. The model will also adapt to changing demands and ensure that the optimal workforce is maintained from hour-to-hour on a daily basis.

The paper is focused on the quantity of alerts investigated by analysts rather than quality of work performed. Hence, the paper did not consider the importance of the alert (category or severity of the alert) because the scope of the paper is capacity building of the cyber-security organization based on the time needed to analyze an alert. It is assumed that when an adequate amount of time is spent on an alert then the alert is thoroughly investigated by an analyst. The subsequent categorization of the alert based on its severity, and the false positive rates (FPR) and false negative rates (FNR) that occur during categorization were beyond the scope of this paper. However, a new and finer notion of risk is to relax the assumption that all alerts that were thoroughly investigated were also accurately investigated. The lack of accuracy leads to FPR and FNR, which impacts the security of the organization. The future work will develop models that include both quantity and quality aspects of alert investigation.

Another interesting future work is to build an adversary cost model for generating a varying rate of significant alerts in order to determine the minimum capacity that an organization must have to analyze all the alerts generated per day. For example, an attacker could know the organization's capacity and attempt to exploit it with a higher rate of alert generation. In such a case, the dynamic model would become useful to bring in additional on-call work force. Also, one can design the organization's capacity to meet a certain threshold in order to better cope with the situation wherein an attacker is attempting to fully exploit the current capacity of the organization. Such a design will make it harder for the attacker to send in enough alerts that will remain unanalyzed, and will limit the attacker's capability to generate several new significant alerts by increasing the cost of such alert generation.

Such an extension that includes a dynamic workforce component or an all proactive dynamic adaptive model, monitoring of analyst performance (FPR/FNR), and incorporating adversary cost modeling for generating a varying rate of significant alerts, is poised to further increase the efficiency of the cybersecurity organization to minimize the overall threat from intrusions that challenge the digital component of every nation's enterprises.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Cliff Wang of the Army Research Laboratory for suggesting this problem to us.

REFERENCES

James P. Anderson. 1980. *Computer Security Threat Monitoring and Surveillance*. Technical Report. James P. Anderson Co., Fort Washington, PA.

Daniel Barbara and Sushil Jajodia (Eds.). 2002. *Application of Data Mining in Computer Security*. Advances in Information Security, Vol. 6. Springer.

CIO Chief Information Officer. 2008. *DON Cyber Crime Handbook*. Dept. of Navy, Washington, DC.

A. D'Amico and K. Whitley. 2008. 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*. 19–37.

Dorothy E. Denning. 1986. An Intrusion-Detection Model. In *Proceedings of IEEE Symposium on Security and Privacy*. Oakland, CA, 118–131.

Dorothy E. Denning. 1987. An Intrusion-Detection Model. *IEEE Trans. Software Eng.* 13, 2 (1987), 222–232.

Roberto Di Pietro and Luigi V. Mancini (Eds.). 2008. *Intrusion Detection Systems*. Advances in Information Security, Vol. 38. Springer.

Haitao Du and Shanchieh Jay Yang. 2013. Temporal and Spatial Analyses for Large-Scale Cyber Attacks. In *Handbook of Computational Approaches to Counterterrorism*, V. S. Subrahmanian (Ed.). Springer, New York, 559–576.

Robert F. Erbacher and Steve E. Hutchinson. 2012. Extending Case-Based Reasoning to Network Alert Reporting. In *2012 ASE International Conference on Cyber Security*. 187–194.

David Goldberg. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, New York, NY.

John R. Goodall, Wayne G. Lutters, and Anita Komlodi. 2004. I know my network: collaboration and expertise in intrusion detection. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*. 342–345.

Donald Gross, John Shortle, James Thompson, and Carl Harris. 2008. *Fundamentals of Queuing Theory*. Wiley-Interscience, New York, NY.

John Holland. 1975. *Adaptations in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

Klaus Julisch and Marc Dacier. 2002. Mining intrusion detection alarms for actionable knowledge. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 366–375.

Stephen Northcutt and Judy Novak. 2002. *Network Intrusion Detection, 3rd Edition*. New Riders Publishing, Thousand Oaks, CA.

Vern Paxson. 1999. Bro: a system for detecting network intruders in real-time. *Computer Networks* 31, 23-24 (1999), 2435–2463.

Michael Pinedo. 2009. *Planning and Scheduling in Manufacturing and Services*. Springer, New York, NY.

Reza Sadoddin and Ali Ghorbani. 2006. Alert Correlation Survey: Framework and Techniques. In *Proceedings of the ACM International Conference on Privacy, Security and Trust. ACM, New York*. 1–10.

Robin Sommer and Vern Paxson. 2010. Outside the Closed World: On Using Machine Learning For Network Intrusion Detection. In *Proceedings of IEEE Symposium on Security and Privacy*. 305–316.

El-Ghazali Talbi. 2009. *Metaheuristics*. Wiley-Interscience, New York, NY.

Fredrik Valeur, Giovanni Vigna, Christopher Kruegel, and Richard A. Kemmerer. 2004. A Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE Transactions on Dependable and Secure Computing 1(3):146-169* (2004).

Wayne Winston. 2003. *Operations Research*. Cengage Learning, New York, NY.

Carson Zimmerman. 2014. *The Strategies of a World-Class Cybersecurity Operations Center*. The MITRE Corporation, McLean, VA.