Mapping Sample Scenarios to Operational Models

Z. Berkay Celik*, Nan Hu*, Yun Li†, Nicolas Papernot*, Patrick McDaniel*, Robert Walls*, Jeff Rowe†, Karl Levitt†, Novella Bartolini*, Thomas F. La Porta*, and Ritu Chadha‡
*Department of Computer Science and Engineering, The Pennsylvania State University
Email: {zbc102, nqh5045, ngp5056, mcdaniel, rjwalls, nub15, tlp}@cse.psu.edu
†Department of Computer Science, University of California, Davis
Email: {yunli, rowe, levitt}@cs.ucdavis.edu
‡Applied Communication Sciences, Basking Ridge, NJ, USA
Email: rchadha@appcomsci.com

***Abstract*—Achieving mission objectives in complex and increasingly adversarial networks is difficult even under the best of circumstances. Currently, there are few tools for reasoning about how to react to rapid changes in a given network's environmental state; that is, we do not know how to cope with adversarial actions in hostile environments. In this paper, we consider a preliminary operational model that combines the states, detection outputs, and agility maneuvers associated with a cyber-operation in hostile networks. The goal is positing the development of an operational model to aid in the successful completion of mission objectives with a minimal maneuver cost. We present a host remediation case study that explores the efficacy of the proposed model in aiding operation completion.**

I. Introduction

The growth and complexity of information systems come with increased necessity for network supervision and administration, so as to prevent the attacks and malicious use of the security and network infrastructure. However, networks or systems constructed in these systems are neither guaranteed to be in a consistent state or to have a uncompromised capability. A lack of knowledge of the states of the infrastructure, users, and adversary creates many challenges.

Many challenges arise in cyber-decision making—what is the best action (maneuver) to take in a given environment based on understanding of the current state (situational awareness) and mission needs? Proper responses to changes in system states, detection outputs, and agility maneuvers are necessary in order to achieve mission objectives. However, today's conventional cyber-defense mostly considers static attack responses in each interval of a finite horizon. In this paper, we ask the essential question *how do defenders select maneuvers to optimize the security outcomes in uncertain environments?* To address this question, we present a graphical representation of an operational model. This representation incorporates nodes (states), transitions, and control variables. We define detection and agility as they pertain to the proposed model. These areas, in concert with the model we have described, form the grounding necessary to reason through a policy that must be created to guide the best action to take for minimizing the overall attack response costs.

The operational model includes the notion of partial neutralization of attacks and estimation of total costs for future decisions from successful, partially successful, and unsuccessful actions of attackers at various attack stages. The goal of the operational model is to take formally into account detection outputs and maneuver capability to enable cost-effective cyber decision-making. The uncertainty of detection systems can trigger unnecessary and costly maneuvers. Thus, a key problem is taking the cost sensitive actions that enhance system security at a minimal cost. This helps us to make real-time decisions that require a keen understanding of the resources required for different maneuvers as well as the relationships and dependencies between the mission controls and cyber assets.

The application of the proposed model is demonstrated in an example of host remediation scenario. We consider a set of plausible threat scenarios and maneuver options. The results help us illustrate the attack stages in more meaningful ways and also provide more insight into the maneuvers. We demonstrate that agility maneuvers are not always the best way of providing strict security, but actions with seemingly poor immediate effects can have better long term ramifications.

II. Operational Model

Our initial model integrates detection output and maneuver costs into a framework for optimal cyber decision-making. A *cyber operation* is an operation performed in the cyberspace domain. Cyber operations include a wide variety of tasks and include both defensive and offensive maneuvers. Cyber operations are defined for a particular scenario. A *scenario* is a high-level description of a mission including a set of actors and goals. The *operational model* of a scenario provides a formal structure for reasoning about detection outputs, agility maneuvers, and security goals.

For illustrative purposes, consider a scenario in which command loses contact with warfighters carrying sensitive devices. Detection mechanisms trigger a cyber operation to secure the devices under potential threat. The goal of the defender is to secure the devices and any impacted cyber

Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-13-2-0045 (ARL Cyber Security CRA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

assets such as internal networks or servers. The state of the environment plays an important role in selecting the appropriate maneuvers. For instance, there are many possible reasons for the loss of communication—such as enemy capture or loss of signal—and each has different risks and responses. Remotely disabling the device reduces the risk of data exfiltration, but it denies the warfighter an important asset; the device may be just out of communication range. Here, the defender must decide the best course of action based on the risks, payouts, and costs. The operational model formalizes this decision process using detection state (situational awareness) and measurements of risk and payout (value to user or network).

The operation model represents a formal specification of a cyber-scenario, *e.g.*, the actions needed to complete an online task as described in the preceding section. We model an operation as a directed graph where the nodes are the states of the operation, and the edges are the state transitions needed to complete the mission. Each transition can represent atomic actions, abstractions for sub-operations or discrete time intervals. However, based on the scenario, a choice of several maneuvers is possible. One cannot predict with certainty the consequence of these maneuvers given the current system state, but may model it as a stochastic event with many possible outcomes. For that reason, we formalize the operational model as discrete-time, finite-horizon Markov Decision Processes (MDP) [1], [2]. This model enables to obtain *maneuver sequences* to make an optimal decision of maneuver costs that terminates operation under various attacks.

Several have modeled operations using business process languages such as business process modeling notation (BPMN) [3], [4]. However, such models are not sufficiently expressive for our analytical needs. Attack models [5] are similar to operation models. While cyber attacks are a necessary component, modeling every possible attack vector is both intractable and unnecessary [3].

Instead, our model considers the effect, timing, and duration of attacks. This formulation enables reasoning about the situation and minimizing the expected maneuver costs to accomplish the scenario. It is not practical to consider only a stable system state by taking the most strict maneuver. While this may put the operation in a secure state, the maneuver costs can be expensive. More specifically, with the recent complex environments and costly maneuvers, a defender needs to analyze the operational behavior for various cases to make optimal future decisions by considering the current view of the scenario. To accomplish this task, it turns out that we need to characterize system states completely, define the maneuver costs and its transitions which we detail next.

III. FORMULATION

In this section, we deal with the problem of enforcing a scenario to behave in the best possible way with a given detection outputs and maneuvers. We discuss some choices for measures of system performance. This is supported by a proof of optimal decision process when a complete analytic solution is possible.

Model Requirements: We consider a model of a system that consists of two main features: (1) a discrete-time dynamic system and (2) a cost function that is additive over time [1], [2]. We define the operational model by the following variables, relations, and probabilities:

$$x_{k+1} = f_k(x_k, u_k, w_k), \qquad k = 0, 1, \ldots, N-1 \qquad (1)$$

where

k	indexes the discrete time,
x_k	is the state of the system that summarizes past information that is relevant for future optimization,
u_k	is the control or decision variable to be selected at time k,
w_k	is a random process, also called disturbance or noise depending on the context,
N	is the horizon length or the number of times control is applied (which is greater than the attack horizon).

and f_k is the function that describes the system and, in particular, the mechanism by which the state is updated.

We define an additive cost function, so that the cost incurred at time k, denoted by $g_k(x_k, u_k, w_k)$, accumulates over time. Thus, the total cost is defined as:

$$g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)$$

where $g_N(x_N)$ is a terminal cost incurred at the end of the process. However, because of the presence of a random parameter, w_k, the cost is a random variable and cannot be meaningfully optimized. We, therefore, formulate the problem as an optimization of the *expected cost*

$$E(g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)),$$

where the expectation is with respect to the joint distribution of the random variables involved. The optimization is over the controls $u_0, u_1, \ldots, u_{N-1}$, and each control, u_k, is chosen based on the current state of the system, x_k. This is called *closed loop* optimization as opposed to *open loop* optimization when all controls have to be decided at once at time 0 without any knowledge of the state of the system at any time later [1].

Mathematically, in closed-loop optimization, we want to find a sequence of functions, $\mu_k, k = 0, \ldots, N-1$, mapping the system state x_k into a control u_k which when applied to the system minimizes the total expected cost. Thus $u_k \leftarrow \mu_k(x_k)$. The sequence $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ is referred to as a *policy* or *control law*.

For each policy, π, the corresponding cost of a fixed initial x_0 is denoted by $J_\pi(x_0)$. We want to minimize this for a given x_0 over all policies that satisfy the constraints of the problem. The policy that does this is denoted by π^* and minimizes the corresponding cost, $J_{\pi^*}(x_0)$. However, it is also possible to find the policy that minimizes the cost of any state.

An introduction to a few notations are in order now. We denote by $J_k(x_k)$ the *cost-to-go* from state x_k at time k to the final state at time N. Thus, $J_N(x_N)$ is the terminal cost and $J_0(x_0) = J_\pi(x_0)$ is the total cost.

Optimal Policy Algorithm: An optimal total cost is given by the last step of the following algorithm, which proceeds backwards in time from period $N-1$ to period 0:

$$J_N(x_N) = g_N(x_N)$$
$$J_k(x_k) = \min_{u_k} \underset{w_k}{E}\Big\{g_k(x_k, u_k, w_k) + J_{k+1}(x_{k+1})\Big\}$$

where $k = 0, 1, \dots, N-1$, and the expectation is taken with respect to the probability distribution of w_k, which depends on x_k and u_k. Furthermore, if u_k^* minimizes the right side of Equation 2 for each x_k and k, the corresponding policy π^* is optimal.

Example: We apply optimal control formulation to devise a minimum cost agility strategy based upon our operational model of a simplified scenario as depicted in Figure 1. For illustration purposes, consider a machine or a network that provides some service. This machine operates by executing a sequence of tasks described as a linear sequence of states with tasks as transitions. Also, this machine is vulnerable to attack, so we also incorporate unknown insecure states; successful attacks cause the system to transition to these unknown insecure states. States corresponding to normal operation of the system are *Good* states, each with corresponding insecure states are *Bad* which we denote as G_k and B_k.

Furthermore, agility maneuvers are transitions that take the machine to some safe state. We use a linear sequence of states, where for each state we have two possible control decisions: *Continue* or *Abort*. For state x_k, we denote these as C_k and A_k respectively. The uncertainty may be incorporated in two ways. For instance, a priori likelihood of being attacked in a given state, which can be derived from the system vulnerabilities. We denote this as λ_k, which is simply the probability of being attacked in state k which results in a transition from G_k to B_{k+1} if the control decision at time k is C_k. If the control decision at time k is A_k, then the task will abort and return to the state G_0 regardless of whether there was an attack or not.

For this example, we assume the system never transitions from a *Bad* to a *Good* state on its own. Only an *Abort* control can recover the system from any *Bad* state back to the *Good* starting point. We then have the following probabilities conditioned on the control decision taken at that discrete time interval, $P(x_{k+1}|u_k)$.

$$P(G_{k+1}|C_k) = (1 - \lambda_k) \qquad P(B_{k+1}|C_k) = \lambda_k$$
$$P(G_0|A_k) = 1 \qquad P(B_{k+1}|B_k) = 1$$

If the information about states were perfect, then the control decision would be trivial. Always *Abort* whenever a *Bad* state is encountered. However, we only have estimates of which state the system is in using an external attack detector which is imperfect. This makes our formulation an optimal control problem with uncertain information. This can be modeled with random variables of detection systems. For instance, the false positive and false negative rates as a vector of attack outputs of the detector. We denote the contents of this vector with detector information at time k as I_k. Note that Information I can be defined at different levels of abstraction as we detail in Section IV.

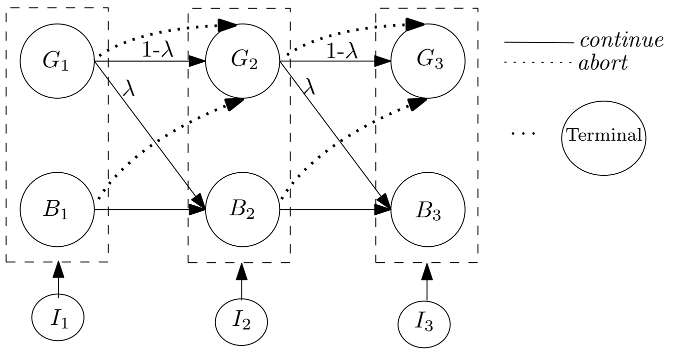


Fig. 1: **The states, control variables and transition probabilities of an operational model:** The model can be extended for various states and control variables considering more complex scenario requirements. This model is evaluated with the host remediation study in Section IV.

The optimal control problem now becomes, given imperfect information (*i.e.*, a vector of attack detection outputs), what control decision will give the lowest expected cost. To do so, the final component of the model is the configuration of cost. We use a simple model with a single cost for the *Abort* control decision, D_A. The cost for *Continuing* and transitioning to one of the *Bad* states is D_k, and the cost for *Continuing* and transitioning to a *Good* state is D_G. In practice, $D_G = 0$ can be chosen since it serves as an overall baseline normalization factor in our simple case. This example model can be generalized to a large number of states, transitions and cost assignments based on the scenario.

Now, to solve the optimal control problem with imperfect information, we need an expression for the recursive cost. By setting the final cost to be arbitrary for an abstract, final state, the recursive formula can be used to work backward in time to generate the optimal control strategy for a given attack detection vector at any point in the system process. This minimal expected cost function, $J_k(I_k)$ depends upon the conditional probabilities described above and takes the following form:

$$\begin{aligned}J_k(I_k) = \min\{ & \\ & (P(G_k|I_k, C_k)D_G + P(B_k|I_k, C_k)D_k + E\{J_{k+1}(I_{k+1})\}, \\ & (P(G_k|I_k, A_k)D_A + P(B_k|I_k, A_k)A_k + E\{J_{k+1}(I_{k+1})\}\}\end{aligned}$$

This provides an expected cost for a given observed attack detection vector. The probabilities in this expression are computed using the detection outputs for each entry in the vector. Selection of the minimum expectation costs provides the optimal control policy for either *Continuing* or *Aborting* back to a stable system state.

IV. Application of Operational Model

To evaluate the operational model, we consider a host remediation scenario in which a set of hosts is infected with malware. An infected host attempts to contact a malicious server for filtering data. The operation uses increasing evidence to determine when to intercede and to perform mitigation on internal networks, servers, and sensitive information.

In this particular scenario, there are many possible explanations for the alarms generated for a host—such as host attempts malicious behavior or false negatives—and each poses different risks and may require a different response. A compromised host may also provide an avenue of cyber-attack for the adversary. Remotely disconnecting the host from network reduces the risk of infection, but it denies the host as an important asset; the host may not be infected, and the alarm may be a false alarm. Other anomalies besides malicious activity may trigger the detection system. For instance, the host may visit some websites that were never visited previously, causing an outlier in the history of the white domain list. Ultimately, when making maneuver decisions, the mission objectives must be balanced against the uncertainty.

Figure 1 presents the transition diagram outlining the scenario. The number of states and control variables are selected to illustrate the process of converting the host remediation into an operational model. The state represents the current environment state and is labeled with the properties that hold in that state with the information provided. In this scenario, we only consider three transitions that trigger the next state operations: whether or not the compromised hosts are (i) operating normally, (ii) quarantined from the rest of the network, or (iii) permanently disabled. Each state represents the states in a discrete time interval, and each transition is associated with a maneuver cost. As mentioned previously, a state transition can only be changed based on the attack probability (λ) obtained from information vector (I). As such, a transition t from state i to state j implies that the state of i meets the requirements of the task and state j is the resulting outcome.

Note that we may be uncertain as to the outcome of a particular attack vector. For example, we have two possible states at the initial state. The a priori likelihood of being attacked in a given state and the decisions that are represented as out-going edges from each state except for final state. Intuitively, this reflects whether or not the control process will proceed with transitions till the termination of the scenario. We next present an implementation of the proposed operational model in a simulated network.

A. Simulation

We first characterize system states, define the maneuver costs and its transitions in a host remediation scenario. Then, we use the operational model to find an optimal attack response for each state of the operation. We have implemented the proposed operational model and a malware scenario in the CyberVAN testbed [6]. CyberVAN is a platform for conducting cyber security experiments. It supports a NS-3 based emulation environment for running real applications on virtual machines. The scenario includes public internet and private enterprise computers. There are also enterprise and public name servers to resolve and register the domain names.

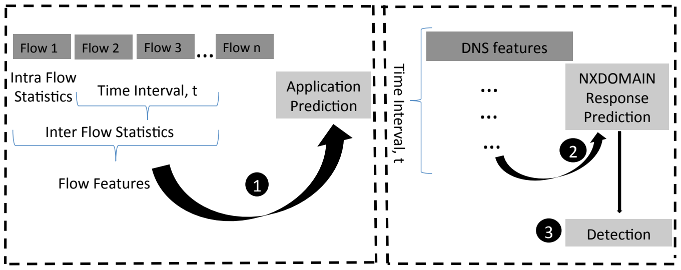


Fig. 2: An overview of proposed malware detection system for host remediation.

The experimental network design includes enterprise hosts that simultaneously generate various types of Markovian traffic such as HTTP, SMTP, and FTP [7]. Each host is infected by the assumption of a host tricked into downloading malware over public internet and executing it. Then, malware generates a domain name generation algorithm (DGA) to communicate with the rendezvous point. Infected hosts are able to generate a range of strings based on the current time and query the public name server for each domain name. If the public name server resolves any of the strings, the hosts use the resolved address as a rendezvous point. The attacker periodically drops rendezvous point and picks a new rendezvous point with the following parameters to increase the granularity of the infected host behavior:

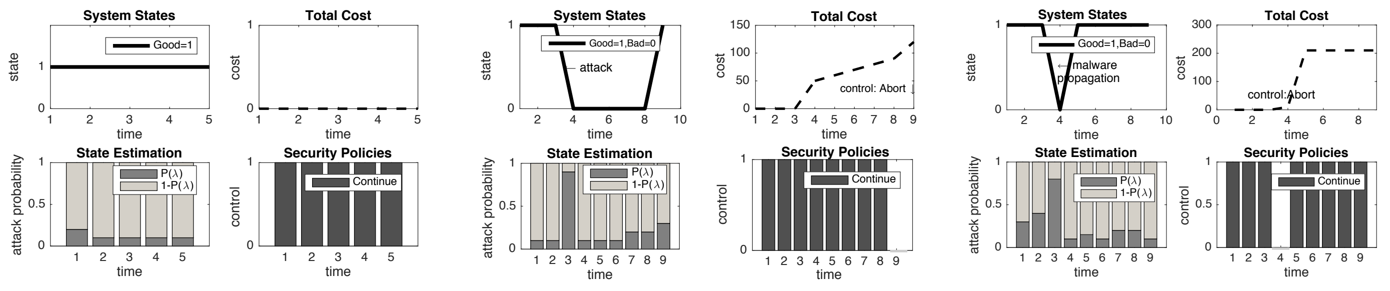
- *Minimum and Maximum Status:* An infected host attempts to send a status message to the C&C server for a random period between min-status and max-status. This results in name server lookup to find the current rendezvous point.
- *Time Range:* This is used to control the number of candidate C&C servers that are generated by the host for sending a status message.

These parameters provide us the control over the number of infected hosts, which results in various detection outputs. Therefore, we can evaluate the scenario under various attack vectors. To characterize the model, we require the attack vector estimations and maneuver cost assignments. We next show how we define these so that we can use the model for minimizing the expected cost of maneuvers.

B. Attack Vector Estimation

The model's first task is to assess the situation. In other words, the system must determine the current state of the scenario. For example, such information (*i.e.*, I vector) may include latest activities of the hosts. Starting from an initial state requires that the detection mechanism has identified some anomaly in the system. We model the attack vector as a probabilistic output of the information vector. Thus, instead of directly observing the current state, the attack vector represents the probability of a host involving a malicious behavior.

A detection system is implemented to infer the behavior of a host that is at imminent risk of infection. In this way, we are able to evaluate the hosts that are contacting a malicious website, communicating with a C&C server, or filtering data



(a) (b) (c)

Fig. 3: **Example of maneuver policies for each time with attack estimations and cost assignments:** a) Model behavior under normal operation: no maneuvers are considered in 5-time intervals (π= {C,C,C,C,C}) b) Attack response with a high probability of high attack estimation at time 3 (π= {C,C,C,A,C,C,C,C,C}) c) Malware infection of multiple hosts at time 1-3 (π= {C,C,C,,C,C,C,C,C,A}). C and A are used for *continue* and *abort* controls, respectively.

to previously unseen external destination. Such actions aimed at revealing the members of a botnet that are controlled by an attacker for malicious activities. The output is used to infer the transitions as a function of system state (s_i) and attack probability (λ_i) at a given time i.

The detection system is built on multiple sources of information that are related to the scenario. Figure 2 presents the three main steps of detection system implementation: from flow traffic to the DNS level. The detection system aims to find the infected hosts with malware that may run applications that generate anomalous traffic and make DNS queries [8]. We use flow statistics to identify the applications, and associate the applications with the corresponding DNS traffic patterns, and then we predict the number of NXDomain (Non-Existent Domain) responses. Finally, we measure the confidence of the actual number of NXDOMAINs on the predicted ones. If the confidence level is low between the predicted and actual number of NXDOMAIN responses, we flag compromised hosts to report a potential security incident. We use the Support Vector Machines (SVM) algorithm for application prediction, and stochastic gradient descent regression to predict the number of NXDOMAIN responses.

The detection output is used to obtain the attack vectors (λ) for the operation of the network in that time interval. This influences the transition probabilities for determining the next states. Given a number of hosts, we can infer the transition probabilities from i to j with a k control by tracking the number of flows associated with each host:

$$\lambda_{i,j,k} = \frac{\sum_{l=1}^{d} f_l}{\sum_{l=1}^{n} f_l} \tag{2}$$

where d represents the number of hosts for which an alarm is generated by the detector and n is the number of total hosts their flows f are processed in a given time interval.

C. Maneuver Cost Assignments

Recall that the main goal of the operational model is minimizing the expected cost of maneuvers under uncertainty. The costs are determined by the attack response taken by the defender and consider the actions taken to stop or remediate the malware. Further, it may be formulated based on the costs defined by equipment, development expenses, and labor, as well as user satisfaction [9].

Specifying the cost function of an operational model is crucial, requiring human assessment of the precise definition of, and tradeoffs among, various states and controls [10]. In the host remediation scenario, we consider three basic response maneuvers that a defender may activate for remediation or removal of the malware. These manage the cost of the state transition and are defined as follows: (i) try to isolate the host or hosts from network, possibly permanently disabling the host (*i.e.*, abort control variable) (ii) quarantine the host to limit risk to other systems or other missions (*i.e.*, continue control variable with bad state transition), or (iii) observe the hosts further before a quarantine is necessary (*i.e.*, continue control variable with good state transition).

We consider a centralized cost assignment that fits the scenario to infer their impact on the operational model control policies. We assign stationary costs for each transition at time i, which corresponds to costs assigned independent of time. We determine the costs as increasing values of defined maneuvers and state transitions, *e.g.*, cost of transiting good in continue control is less than abort control, as abort requires more work in isolating and removing the malware. We also foresee more advanced cost assignments. For instance, a dynamic cost function that assigns costs based on a function of attack vector estimation outputs. This algorithm trusts the detection outputs as a basis. If detection systems indicate high levels of malicious activity, the remediation of hosts may incur more cost. The cost assignments can be generalized to a broader definition of the scenario and corresponding random or intelligent moving target defense configurations [11].

V. Experimental Results

We first present examples of the operational model to understand better its capabilities. Figure 3 illustrates the examples of states, their transitions, control variables, associated costs and previously observed detection outputs.

Figure 3a illustrates the case of no malicious activity in the network. This yields a detector generating perfect knowledge,

	t_1	t_2	t_3	t_4	t_5
Case 1	0.2	0.3	0.5	0.8	terminal
Case 2	0.2	0.9	0.8	0.1	terminal

TABLE I: Attack probabilities (λ) governing state transitions

State Transition	**Cost**	**Control**
$D_{G_iG_j}$	0	Continue
$D_{G_iB_j}$	10	Continue
$D_{B_iB_j}$	15	Continue
D_{B_i} and D_{G_i}	20	Abort

TABLE II: Cost assignments of transitions for each operational model control policy

i.e., the information vector produces almost 0% detection error. By convention, we only observe properties that hold in each state. As a result, the operation is terminated at the cost of 0, as no additional maneuvers are considered.

Second, we introduce an attack at time interval $t = 3$ as presented in Figure 3b. This results in successful infection of a few hosts with malware. The model transits to a bad state with the control variable $continue$. At time $t = 8$, we observe that cost becomes 100 after the appropriate maneuvers are taken. Then, the system switches to good state by the $abort$ control variable with a total cost of 120. However, if the system state is changed to abort earlier, the cost increases, because aborting and starting from the good state then switching to bad state and aborting again incurs additional costs. This sequence eliminates the optimal control decisions. For this particular case, we found that keeping the system in the bad state for a limited time is the best decision based on the maneuver costs compared to aborting and recovering back to the good state.

Finally, Figure 3c presents malware propagation that results in multiple host infection. The infection occurs at time $t = 1 - 3$ with a high attack probability. In this case, a high-cost optimal maneuver is taken to put system state in the good state. The reason is to ensure that once a quarantined host resumes communication, the malicious software must be removed before continuing the communication. This represents the best control policy to minimize the overall maneuver costs.

Now, we evaluate the operational model with previously obtained detection outputs. Table I presents the two different attack probabilities of five minute time intervals, $t = 0 - 5$ and so on. Table II presents the costs assignments for each transition. The values are assigned considering the maneuvers as described in Section IV-C. The goal of the model is to lead the system to a good state at the terminal state by considering $continue$ and $abort$ controls. We now apply the optimal control model. The solution of the operational model outputs a policy that specifies the best control to take for each state.

Table III presents the optimal controls obtained for each state of $good$ and bad at time t_i to successfully terminate the operation. For instance, in case 1, the system uses $abort$ control in Bad (s_2) at time t_1 and t_2 . However, in case 2, controls policy changes, and even we observe $abort$ control in good state. This is because the detection system reports a high number of infected machines, as the attack probability increases, the system attempts to abort to put the system in good state to minimize the attack response. These results show

		t_1	t_2	t_3	t_4	t_5
Case 1	$s_{1,t}$	Continue	Continue	Continue	Continue	Continue
	$s_{2,t}$	Abort	Abort	Continue	Abort	Continue
Case 2	$s_{1,t}$	Continue	Continue	Abort	Continue	Continue
	$s_{2,t}$	Abort	Continue	Abort	Abort	Continue

TABLE III: Optimal control policies (π) obtained for each state with the values in Table I and II.

the optimal decisions for each state, and states that if any other control sequences are considered, the policy will be more costly or the scenario will fail to terminate in good state.

Finally, we note that operational model enables various assignments of the optimal policy of variables: system states ($s_{ij}, \ldots, s_{in}$), attack vectors in following states ($\lambda_1, \ldots, \lambda_n$), and maneuver cost assignments for each transition or state ($Dij, \ldots, D_{in}$). The model can be used with various values of the variables to obtain the optimal attack response policies. Building from this model, future work will consider more states with their respective control loops and integration of risk in transitions. From this basis, the model above may mature.

VI. Conclusion

Networks and systems are growing evermore complex. With more complexity comes more difficulty in enforcing an optimal attack response policy with various maneuvers. Currently, there is no standardized model for reasoning about how to react to rapid changes in a given threat. In this paper, we presented an operational model that combines various aspects associated with the completion of a cyber-operation. This is necessary for the development of autonomic systems to aid in the successful enforcement of a maneuver policy. We presented a scenario to show the efficacy of the operational model in aiding the analysis of host remediation.

References

[1] D. P. Bertsekas, *Dynamic programming and optimal control.* Athena Scientific, 2005.
[2] D. P. Bertsekas and S. E. Shreve, *Stochastic optimal control: The discrete time case.* Academic Press, 2007.
[3] A. de Barros Barreto, P. C. G. da Costa, and E. T. Yano, "Using a semantic approach to cyber impact assessment." in *STIDS*, 2013, pp. 101–108.
[4] S. Jajodia, A. K. Ghosh, V. Subrahmanian, V. Swarup, C. Wang, and X. S. Wang, "Moving target defense ii," *Application of game Theory and Adversarial Modeling. Series: Advances in Information Security*, 2013.
[5] G. Jakobson, "Mission cyber security situation assessment using impact dependency graphs," in *Information Fusion.* IEEE, 2011.
[6] "Cyber virtual ad hoc network (cybervan)," http://www.appcomsci.com/research/tools/cybervan, [Online; accessed 10-September-2015].
[7] A. Nogueira, P. Salvador, R. Valadas, and A. Pacheco, "Markovian modelling of internet traffic," in *Network performance engineering.* Springer, 2011, pp. 98–124.
[8] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. Dagon, "From throw-away traffic to bots: detecting the rise of dga-based malware," in *USENIX Security Symposium*, 2012, pp. 491–506.
[9] W. Lee and D. Xiang, "Information-theoretic measures for anomaly detection," in *Security and Privacy.* IEEE, 2001, pp. 130–143.
[10] K. Regan and C. Boutilier, "Eliciting additive reward functions for markov decision processes," in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 22, no. 3, 2011, p. 2159.
[11] R. Zhuang, S. Zhang, A. Bardas, S. A. DeLoach, X. Ou, and A. Singhal, "Investigating the application of moving target defenses to network security," in *International Symposium on Resilient Control Systems (ISRCS)*, 2013.