2019 American Control Conference (ACC)
Philadelphia, PA, USA, July 10-12, 2019

Optimal Reset Strategies for Mitigating Malware Epidemics

Nicholas J. Watkins and George J. Pappas

Abstract— **As the size of the Internet of Things (IoT) grows, so does its vulnerability to malicious attacks. When such an attack occurs, one of the only means of defense available to a system controller before a patch is developed is resetting devices to a known malware-free state. In this paper, we study the design of reset strategies which optimize the network's performance when under a malware attack. In particular, we show that under mild assumptions, the problem of optimizing the network's performance can be posed as an optimal control problem for a Markov chain with a number of states and actions which grows polynomially with respect to the size of the network. We investigate our results with simulation.**

I. Introduction

As we consider building an Internet-of-Things (IoT), we must be conscious of the difficulty of ensuring its secure operation. Already, we have seen several large scale cyberattacks have significant effects on the operation of the internet for prolonged periods of time [1]–[3]. This problem will become worse as more devices are added to the IoT.

Due to the complexity associated with designing secure devices, and the economics of producing components at a price point where they can be widely adopted, we can reasonably expect any device used as a node in the IoT to be manufactured with inherent security flaws [4]. Certainly, once a security flaw is noticed by a manufacturer's maintenance team, it is plausible that a software patch will be developed and disseminated among affected devices. There are many works which study methods for optimizing patch dissemination in this setting (see, e.g., [5]–[9]). However, it is often the case that much damage is done by a malware *before* the security exploit used in the attack can be identified by the manufacturer's response team - this is why zero day exploits carry significant financial value [10], [11].

As such, it is important to develop response methods for protecting against malware attacks using IoT devices which operate without the capability of patching nodes to inoculate them against future reinfection. To accomplish this, we must consider leveraging features of malware which are commonplace, and can be used as control levers to mitigate the effect of a malware outbreak. One such feature is the widespread vulnerability of malware to *device resets* (consider, e.g., the malware Mirai and its variants [1]–[3]).

While promising in concept, it is not obvious that device resets can be used effectively as a means for responding to a malware attack. For one reason, device resets take non-negligible time, and remove resources from the network while they are taking place. Additionally, a reset will not protect a device from reinfection with whatever malware had afflicted it in the first place, as the security exploit used has not been identified or patched. As such, it is not obvious when a decision to reset a device should be made, or whether such an approach can be effective. Indeed, it is not immediately obvious if *optimal* reset strategies can be efficiently computed, as models for malware are complicated, and the number of ways devices can be assigned to reset grows combinatorially with the size of the network. We study techniques for computing optimal reset policies in this paper.

Statement of Contributions: Whereas prior works studying device reset as a means of malware control [12], [13] have designed heuristic strategies, we believe this paper is the first to study the design of *optimal* reset strategies for defending networks against malware attacks. As core technical contributions, we show that under mild assumptions, searching for an optimal reset strategy can be posed as a Markov decision process on state and action spaces which grow polynomially with respect to the number of devices in the network to be controlled, and that this low-complexity problem representation can be computed efficiently from the problem's specification. This result is novel, and its derivation requires an analysis which we believe will be of interest broadly to epidemic control researchers, as the arguments required are not context dependent.

We investigate the utility of our results with simulations. Moreover, the performance of optimal policies is found to be substantially better than that of heuristic policies, suggesting that developing efficient methods for optimizing malware response policies is a fertile area for future research.

Organization of Remainder: The remainder of the paper is organized as follows. Section II contains a detailed epidemic model for computer malware, as well as a formal technical statement of the problem we address in the paper. Section III provides a novel model reduction technique useful for computing optimal reset policies aimed at mitigating the effects of a malware attack. Section IV provides an example application, in which the developed model reduction technique is shown to enable the efficient computation of a reset policy which optimizes a network's performance when responding to a malware attack. •

Notation and Terminology: We denote by $[k]$ the set of the first k positive integers, i.e. $[k] \triangleq \{1, 2 \ldots, k\}$. We denote by $[k]_0$ the set of the first $k+1$ natural numbers, i.e. $[k]_0 = \{0, 1, 2, \ldots, k\}$. We denote by $\mathbb{I}_{\{\cdot\}}$ a $0/1$ indicator function, which takes the value 1 if the argument is true, and 0 otherwise. We denote by $\mathbb{E}$ the expectation operator, and by $\mathbb{E}_\xi$ the expectation with respect to a probability measure ξ. Note that when clear from context, we omit explicit reference to the probability measure used. •

II. Model and Problem Statement

In this section, we provide details of the malware model we study, and formally state the problem examined in the paper. Section II-A provides a technical description of the network model. Section II-B provides a technical description of the malware model. Section II-C provides a technical description of the network's utility model. Section II-D provides a formal problem statement.

A. Network Model

We consider the case of malware propagating on a network of n_d interconnected devices, which can be decomposed into

978-1-5386-7928-9/$31.00 ©2019 AACC

n_g groups of statistically identical devices, each of which may take an arbitrary size, where we notate the particular group g's size as n_{dg}, and the set of all groups as G. This assumption seems reasonable for many practical scenarios. Malware programs often propagate on groups of networks of old, out-of-date network devices (e.g. routers, security cameras) such that all devices within each group have similar networking capabilities and security vulnerabilities.

In order to coordinate their actions, we assume all devices communicate with a central controller. At each time step, every device runs a malware detection routine to asses whether or not it is currently infected, and reports the result of its test to the central controller. The central controller then determines which devices, if any, should initiate a reset, with the decision being made to optimize the network's performance (details regarding evaluation of the network's performance are given in Section II-C and Section II-D).

B. Epidemic Model for Mirai-like Malware

As is standard in the study of epidemic processes [14]–[16], we model the current status of a node by membership in one of a set of *compartments*, each of which are labeled with a particular symbol $\ell \in \mathcal{L}$. We denote the indicator that a particular node i is in a particular compartment ℓ with the $0/1$ random variable X_i^ℓ, which jointly satisfy $\sum_{\ell \in \mathcal{L}} X_i^\ell = 1$ for all nodes i, as each node belongs to exactly one compartment at every time. As Mirai and its descendants can reinfect devices after they have been reset [1]–[3], we model it by a discrete-time Susceptible-Infected-Removed-Susceptible ($SIRS$) epidemic. We label nodes which are not currently infected with malware with S (for Susceptible), nodes which are currently infected with malware with I (for Infected), and nodes which are currently undergoing a device reset with R (for Removed).

Nodes transition between membership in different compartments due to the occurrence of random events and applied control actions. We denote by random variables Y_{ij} the occurrence of a contact between node i and node j which will spread malware to node i if node j is infected. We denote by random variables $Z_i^{\ell \to \ell'}$ internal random events which cause node i to transition from compartment ℓ to compartment ℓ'. For example, the random variable $Z_i^{R \to S}$ details whether or not a device currently undergoing a reset comes back online at the next time increment. We model the action of forcing a node i to initiate a reset with the control variable $a_i \in \{0, 1\}$, where $a_i = 1$ indicates that node i is forced to initiate a reset, and $a_i = 0$ the opposite. Note that since devices which are currently undergoing a reset cannot communicate with the central controller, the particular choice of a_i has no effect on nodes which are currently in compartment R.

Assembling this notation in to a mathematical model, we have that node i's indicator for compartment S evolves as

$$X_i^{S+} = (1 - a_i)(W_i(X) X_i^S + X_i^I Z_i^{I \to S}) + X_i^R Z_i^{R \to S}, \quad (1)$$

where by $W_i(X)$ we denote a $0/1$ random variable indicating whether a neighbor of node i has attempted to infect it with malware. More precisely, we have

$$W_i(X) \triangleq (1 - Y_{i0}) \prod_{j \in [n_d]} (1 - Y_{ij} X_j^I), \quad (2)$$

where we treat the random variables Y_{i0} as infection attempts generated by the attacker (assigned the label of node 0). Note that $W_i(X)$ takes the value 1 if no successful infection attempt as been made, and 0 otherwise. In a similar fashion, we may write the dynamics of the indicator of infection as

$$X_i^{I+} = (1 - a_i)((1 - W_i(X)) X_i^S + (1 - Z_i^{I \to S}) X_i^I), \quad (3)$$

and the indicators for reset states as

$$X_i^{R+} = a_i (X_i^S + X_i^I) + X_i^R (1 - Z_i^{R \to S}). \quad (4)$$

In Section II-A, we made the assumption that the network is comprised of a collection of groups of statistically identical devices. As such, this places some restrictions on the distributions of the random variables which comprise the process. In particular, we have that for any two groups g and g' in G, if nodes i and k are both elements of group g and nodes j and h are elements of group g', then $\Pr(Y_{ij} = 1) = \Pr(Y_{kh} = 1)$, i.e. the distributions of Y_{ij} and Y_{kh} are identical. Likewise, we have that for all nodes i in a particular group g, the random variables which indicate internal transitions (i.e. $Z_i^{\ell \to \ell'}$) are identically distributed.

Finally, we note that all random variables which indicate the occurrence of events which cause compartmental transitions (i.e. Y_{ij} or $Z_i^{\ell \to \ell'}$) are assumed to be mutually independent at all times. This is not a strong assumption, as the elements of the network state process $\{X(t)\}$ remain strongly correlated, both across devices in the network and across time. Note also that this assumption makes practical sense as well, as it suggests that each device's inherent properties (e.g. reset time) and the malware's inherent properties (e.g. attack strategy) remain fixed in time.

In all, we can see that the process $\{X(t)\}$ is a controlled Markov chain which evolves on a state space $\mathcal{X}$ with 3^{n_d} elements, on which 2^{n_d} possible actions $a \in \mathcal{A}$ can be applied. Note that each state $X \in \mathcal{X}$ corresponds to one particular combination of compartmental memberships, and each action $a \in \mathcal{A}$ corresponds to one particular assignment of reset initiations. The principle difficulty in controlling $\{X(t)\}$ efficiently is due to the large size of its state space, and the large set of possible actions which can be applied. While in general it may not be possible to compute an efficient control policy, in Section II-C we define a large class of utility functions for which we demonstrate that optimal control policies can be computed in polynomial time (using methods developed in Section III).

C. Utility Model

We assume that the network's manager provides the controller with some function $\mathcal{U}$ which maps the current state X of the network to a finite, non-negative number corresponding to the amount of utility derived from the network when it is in state X. For example, if we consider our network as being a swarm of several types of robots tasked with surveying and defending an area, we should only care that the total performance of the team is maximized. As such, we may chose $\mathcal{U}(X) = \sum_{i \in [n_d], \ell \in \mathcal{L}} \alpha_{g(i)\ell} X_i^\ell$, where $\alpha_{g\ell}$ is some nonnegative constant denoting how much utility can be derived from a device of type g currently in compartment ℓ, and we use the notation $g(i)$ to denote a function which maps the index i to its group membership (in this example, the type of robot of device i). Similarly, if

we are concerned with responding to a Distributed Denial-of-Service (DDoS) attack, we should chose $\mathcal{U}$ to ensure that the amount of traffic experienced by the targeted server is below its maximum operating capacity, while also attempting to keep as many network devices operational as possible. We can accomplish this by choosing

$$\mathcal{U}(X) = \mathbb{I}_{\{\sum_{i\in[n_d],\ell\in\mathcal{L}} \beta_{g(i)\ell} X_i^\ell \leq \theta\}} \sum_{i\in[n_d],\ell\in\mathcal{L}} \alpha_{g(i)\ell} X_i^\ell,$$

where $\mathbb{I}_{\{\cdot\}}$ is a $0/1$ indicator function, $\beta_{g\ell}$ is the amount of traffic generated by a device in group g in compartment ℓ, and θ is the targeted server's traffic capacity.

Notice that these two choices of utility functions exhibit a particular kind of symmetry, which we refer to as *count symmetry*. More precisely, let $\mathcal{C}$ be the vector function

$$\mathcal{C}_{g\ell}(X) \triangleq \sum_{i\in g} X_i^\ell, \tag{5}$$

i.e. $\mathcal{C}_{g\ell}(X)$ denotes the count of the devices in group g currently in compartment ℓ of the malware model. If it holds that $\mathcal{U}(Y) = \mathcal{U}(Z)$ for all Y and Z in $\mathcal{X}$ such that $\mathcal{C}_{g\ell}(Y) = \mathcal{C}_{g\ell}(Z)$ holds for all device groups g and every compartmental label ℓ, we say that $\mathcal{U}$ is count symmetric. As can be seen from the examples above, count symmetry can be found in practice where groups of statistically identical devices are involved. This makes such functions an appropriate object of study; we assume $\mathcal{U}$ is count symmetric in the sequel.

D. Problem Statement

In this paper, we concern ourselves with optimizing the λ-discounted expected return of the control policy π,

$$J_{\pi,\lambda}(X) \triangleq \mathbb{E}_\pi \sum_{\tau=0}^{\infty} \mathcal{U}(X)\lambda^\tau, \tag{6}$$

where $\lambda \in (0,1)$, and π is a non-anticipating control policy which maps observations of the process state $\{X(t)\}$ to device reset actions $a \in \mathcal{A}$. We consider the problem of computing an optimal reset policy π, i.e. the solution to the optimal control problem

$$\max_{\pi\in\Pi} J_{\pi,\lambda}(X), \tag{7}$$

for all states $X \in \mathcal{X}$, where Π is the set of all non-anticipating control policies which map observations of X to device reset actions $a \in \mathcal{A}$, and we assume the utility function $\mathcal{U}$ used to define J is count symmetric (as detailed in Section II-C).

Because $\{X(t)\}$ is a Markov chain we control by applying one of a finite set of actions $\mathcal{A}$, and we observe the states and utility, the optimal control porblem (7) is a Markov decision process (see, e.g., [17, Chapter 3] for background). As such, there are standard tools for computing a solution, both in the case where the transition probabilities are known (e.g. value iteration; see, e.g., [18, Chapter 1]), and when they are not (e.g. Q-learning; see, e.g., [18, Chapter 6]). However, the complexity of these methods are a function of the size of the process' state and action space. Since $\{X(t)\}$ evolves on a state space with 3^{n_d} elements and we may apply 2^{n_d} different actions at each state, it should be anticipated that (7) is in general difficult to solve. As such, the body of the paper is devoted to studying whether or not the structural assumptions placed on $\{X(t)\}$ and $\mathcal{U}$ in Sections II-A through II-C suffice to make (7) efficiently solvable. We develop a model reduction technique in Section III that enables (7) to be solved in polynomial time. Using the developed model reduction technique, we study the behavior of optimal reset policies in Section IV by simulation.

III. A Model Reduction Technique

In this section, we demonstrate that (7) can be solved in polynomial time. Principally, we proceed by developing a model reduction technique. We demonstrate how to compute a *lumped* representation of (7), such that a solution from the lumped problem can be used to compute a solution to (7) itself. To gain an intuition for why this might be possible, note that the count symmetry of $\mathcal{U}$ implies the existence of a function $\mathcal{V}$ which maps the vector $\mathcal{C}(X)$ to $\mathcal{U}(X)$. Moreover, because the devices within each group are statistically identical, we might believe that $\{\mathcal{C}(X(t))\}$ is itself a controlled Markov process, where actions can be modeled as specifying the number of devices to be reset in each group and compartment, instead of identifying each device to be reset explicitly by its label. It happens to be the case that this intuition is correct.

To be more precise, let $r^\nu_{g\ell}(X)$ denote the number of devices of group g in compartment ℓ that are forced to initiate a reset when the network is in state X under the reset policy ν. Define $e^\nu_{g\ell}(X)$ as the number of devices in group g and compartment ℓ which are eligible to undergo a compartmental membership transition due to some stochastic event (as opposed to being forced to undergo a reset), i.e.

$$e^\nu_{g\ell}(X) \triangleq \mathcal{C}_{g\ell}(X) - r^\nu_{g\ell}(X). \tag{8}$$

The count of susceptible devices in group g then evolves as

$$\mathcal{C}_{gS}(X^+) = \sum_{k=1}^{e^\nu_{gS}(X)} \tilde{W}_{gk}(\mathcal{C}(X)) + \sum_{k=1}^{e^\nu_{gI}(X)} \tilde{Z}^{I\to S}_{gk} + \sum_{k=1}^{\mathcal{C}_{gR}(X)} \tilde{Z}^{R\to S}_{gk}, \tag{9}$$

where the random variables $\{\tilde{W}_{gk}(\mathcal{C}(X))\}$ form a collection of independent, identically distributed random variables which have the same distribution of $W_i(X)$, for $i \in g$, and the random variables $\{\tilde{Z}^{\ell\to\ell'}_{gk}\}$ form a collection of independent, identically distributed random variables with the same distribution of $Z_i^{\ell\to\ell'}$ for $i \in g$. Note that it is possible to construct $\{\tilde{W}_{gk}(\mathcal{C}(X))\}$, since $W_i(X)$ can be written as a function of $\mathcal{C}(X)$ by setting

$$W_i(\mathcal{C}(X)) = (1 - Y_{i0}) \prod_{g\in\mathcal{G}} \prod_{k=1}^{\mathcal{C}_{gI}(X)} (1 - \tilde{Y}_{ig,k}), \tag{10}$$

where when $\mathcal{C}_{gI}(X) = 0$, we take $\prod_{k=1}^{\mathcal{C}_{gI}(X)}(1 - \tilde{Y}_{ig,k}) = 1$, and we define the random variables $\{\tilde{Y}_{ig,k}\}$ as a collection of identical, independent random variables taking the same distribution of Y_{ij} for $j \in g$. Likewise, the dynamics of $\mathcal{C}_{gI}(X)$ can be written as

$$\mathcal{C}_{gI}(X^+) = \sum_{k=1}^{e^\nu_{gS}(X)} 1 - \tilde{W}_{gk}(\mathcal{C}(X)) + \sum_{k=1}^{e^\nu_{gI}(X)} 1 - \tilde{Z}^{I\to S}_{gk}, \tag{11}$$

and the dynamics for $\mathcal{C}_{gR}(X)$ can be written as

$$\mathcal{C}_{gR}(X^+) = r^{\nu}_{gS}(X) + r^{\nu}_{gI}(X) + \sum_{k=1}^{\mathcal{C}_{gR}(X)} 1 - \tilde{Z}^{R\to S}_{gk}. \quad (12)$$

By inspection of (9)-(12), it can be seen that $\{\mathcal{C}(X)\}$ is a function only of number of devices forced to initiate a reset at any particular state $\mathcal{C}(X)$, and not the particular identity of the devices involved. This suggests that we can compute an optimal control policy which acts on observations of states $X \in \mathcal{X}$ and gives actions $a \in \mathcal{A}$ by computing an optimal control policy μ which acts on observations of compartmental membership counts $\mathcal{C}(X)$, and gives actions which specify only the *number* of devices to be reset in each group and compartment, as opposed to the particular devices.

This is useful, because the set of all possible compartmental membership counts has substantially fewer than 3^{n_d} elements, and the number of different device counts to be reset can be selected is substantially smaller than 2^{n_d}. To be more precise, define $\mathcal{V}$ as a function which maps $\mathcal{C}(X)$ to $\mathcal{U}(X)$. Consider the lumped optimal control problem

$$\max_{\mu\in\mathcal{M}} V_{\mu,\lambda}(\mathcal{C}(X)), \quad (13)$$

with $V_{\mu,\lambda}(X) \triangleq \mathbb{E}_{\mu}\sum_{\tau=0}^{\infty}\mathcal{V}(\mathcal{C}(X))\lambda^{\tau}$, and where $\mathcal{M}$ is the set of all non-anticipating control policies which map $\mathcal{C}(X)$ to a particular count of devices to be reset in each group, compartment pair. Optimal solutions of (13) can be used to compute an optimal solution of (7), as the following certifies.

Theorem 1 (Equivalence of Solutions) *Let $\mu^\star$ be an optimal solution to* (13)*. Then, the policy $\pi_{\mu^\star}$, which upon observing $\mathcal{C}(X)$ assigns exactly $\min\{r^{\mu}_{gS}(\mathcal{C}(X)), \mathcal{C}_{gS}(X)\}$ susceptible devices of group g to be reset uniformly at random and exactly $\min\{r^{\mu}_{gI}(\mathcal{C}(X)), \mathcal{C}_{gI}(X)\}$ infected devices of group g to be reset uniformly at random for each group $g \in \mathcal{G}$, is an optimal solution of* (7)*.*

Since (13) is a Markov decision process, it can be solved in polynomial time with respect to the number of states of the system, and actions in the action set (see, e.g., [19, Theorem 1]), provided the transition probabilities are known. This is important, because the complexity of representing (13) is substantially less than the complexity of representing (7). To demonstrate this precisely, consider the following result:

Lemma 1 (A Combinatorial Identity) *Let $\Phi_p(m)$ be the set of all combinations of exactly p natural numbers which sum to m, and let $\phi_p(m)$ denote the cardinality of $\Phi_p(m)$. We have*

$$\phi_p(m) = \frac{(m+p-1)!}{m!(p-1)!}. \quad (14)$$

Moreover, for each fixed p, $\phi_p(m)$ grows as $O(\frac{m^{p-1}}{(p-1)!})$.

Since for each group g, the count of nodes taking membership in each of the three model compartments must equal n_{dg}, we have that $\mathcal{C}(X)$ evolves on the state space $\times_{g\in\mathcal{G}}\Phi_3(n_{dg})$. From Lemma 1, this set contains exactly $\times_{g\in\mathcal{G}}\frac{(n_{dg}+2)!}{n_{dg}!2!}$ elements, and as such grows as $O({n_d}^{2n_g})$. This grows polynomially for a fixed number of device groups, as opposed to $\mathcal{X}$, which grows as 3^{n_d}.

Likewise, we may use $\Phi_3(n_{dg})$ to encode all possible ways of assigning devices to be reset in a particular group g. In particular, for any $\mathcal{C}(X)$ we must have that $r^{\mu}_{gS}(\mathcal{C}(X)) + r^{\mu}_{gI}(\mathcal{C}(X)) \leq n_{dg}$, as there are only n_{dg} devices in group g total. As such, for any particular $d \in \Phi_3(n_{dg})$, we can assign $d_1 = r^{\mu}_{gS}(\mathcal{C}(X))$, $d_2 = r^{\mu}_{gI}(\mathcal{C}(X))$, and $d_3 = n_{dg} - r^{\mu}_{gS}(\mathcal{C}(X)) - r^{\mu}_{gI}(\mathcal{C}(X))$. Hence, the set of possible actions of (13) grows as $O({n_d}^{2n_g})$, which is substantially less than 2^{n_d} elements needed to represent $\mathcal{A}$.

Considering this, it seems that (7) may itself be solvable in polynomial time. Indeed, all that is left to show to demonstrate this is an efficient method of constructing the transition probabilities $\Pr(\mathcal{C}(Y)|\mathcal{C}(X), d)$ for for all $d \in \mathcal{D} \triangleq \times_{g\in\mathcal{G}}\Phi_3(n_{dg})$. This is not trivial. Indeed, a naïve approach to this computation would involve summing over the exponentially large set of possible events which could cause the transition $\mathcal{C}(X) \to \mathcal{C}(Y)$ when the action d is applied. Our next result demonstrates that, provided the distributions (i.e. success probabilities) of the contact random variables $\{Y_{ij}\}$ and internal transition random variables $\{Z_i^{\ell\to\ell'}\}$ are known for each group, then each entry to the table of values for $\Pr(\mathcal{C}(Y)|\mathcal{C}(X), d)$ can be computed in polynomial time.

Proposition 1 (Computing Transition Probabilities) *Consider the model detailed in Section II-B. If the distributions for the random variables $\{Y_{gg'}\}$ and $\{Z_g^{\ell\to\ell'}\}$ (denoted $P_{Y_{gg'}}$ and $P_{Z_g^{\ell\to\ell'}}$, respectively) are known for each $g, g' \in \mathcal{G}$ and $\ell, \ell' \in \mathcal{L}$, then each transition probability $\Pr(\mathcal{C}(Y)|\mathcal{C}(X), d)$ can be computed in $O(n_g{n_d}^6)$ time. Consequently, all such probabilities can be computed in $O(n_g{n_d}^{6n_g+6})$ time.*

Proof: Since the transitions of nodes within each group are conditionally independent given the counts of compartmental memberships of nodes in all groups of the graph, we can decompose the probability into the product

$$\Pr(\mathcal{C}(Y)|\mathcal{C}(X), d) = \prod_{g\in G}\Pr(\mathcal{C}_g(X) \to \mathcal{C}_g(Y)|\mathcal{C}(X), d).$$

Hence, if we can compute the values $\Pr(\mathcal{C}_g(X) \to \mathcal{C}_g(Y)|\mathcal{C}(X), d)$ in polynomial time, we can compute $\Pr(\mathcal{C}(Y)|\mathcal{C}(X), d)$ in polynomial time as well. To demonstrate how this is possible, we decompose the event $\{\mathcal{C}_g(X) \to \mathcal{C}_g(Y)\}$.

If the action d is applied, then $\min\{d_{gS}, \mathcal{C}_{gS}(X)\}$ susceptible nodes and $\min\{d_{gI}, \mathcal{C}_{gI}(X)\}$ infected nodes of group g are forced to initiate a reset. Hence, there are $c_{XS} \triangleq \mathcal{C}_{gS}(X) - \min\{d_{gS}, \mathcal{C}_{gS}(X)\}$ susceptible nodes, $c_{XI} \triangleq \mathcal{C}_{gI}(X) - \min\{d_{gI}, \mathcal{C}_{gI}(X)\}$ infected nodes, and $c_{XR} \triangleq \mathcal{C}_{gR}(X)$ removed nodes from group g which will make a stochastic transition. To result with a node count $\mathcal{C}_g(Y)$ after the transition occurs, we must have that the number of nodes which transition to susceptibility stochastically to equal $c_{YS} \triangleq \mathcal{C}_{gS}(Y)$, the number of nodes which transition to infected stochastically to equal $c_{YI} \triangleq \mathcal{C}_{gI}(Y)$, and the number of nodes which transition to recovered stochastically to equal $c_{YR} \triangleq \mathcal{C}_{gR}(Y) - \min\{d_{gS}, \mathcal{C}_{gS}(X)\} - \min\{d_{gI}, \mathcal{C}_{gI}(X)\}$.

Let $\Psi_{g|d}(X, Y)$ be the subset of $\mathbb{Z}_{\geq 0}^{\mathcal{L}\times\mathcal{L}}$ such that $\sum_{\ell'\in\mathcal{L}}\psi_{\ell\ell'} = c_{X\ell}$ holds for all $\ell \in \mathcal{L}$ and $\sum_{\ell\in\mathcal{L}}\psi_{\ell\ell'} = c_{Y\ell'}$ holds for all $\ell' \in \mathcal{L}$. So defined, each ψ in $\Psi_{g|d}(X, Y)$ defines one way for the random transitions to occur, such that the transition $\mathcal{C}_g(X) \to \mathcal{C}_g(Y)$ occurs. As such, we have

$$\{\mathcal{C}_g(X) \to \mathcal{C}_g(Y)\} = \cup_{\psi\in\Psi_{g|d}(X,Y)}\{\cap_{\ell,\ell'\in\mathcal{L}}\psi_{\ell\ell'}\}, \quad (15)$$

where we use the shorthand notation $\psi_{\ell\ell'}$ for the event that exactly $\psi_{\ell\ell'}$ devices currently in compartment ℓ transition to compartment ℓ' due to random events. Since the elements of $\Psi_{g|d}(X,Y)$ are disjoint, we have by additivity that

$$\begin{aligned}&\Pr(\mathcal{C}_g(X) \to \mathcal{C}_g(Y)|\mathcal{C}(X), d)\\ &= \sum_{\psi \in \Psi_{g|d}(X,Y)} \Pr(\cap_{\ell,\ell' \in \mathcal{L}} \psi_{\ell\ell'}|\mathcal{C}(X)),\end{aligned} \tag{16}$$

where we have dropped the dependence on d in the right-hand side expression to emphasize that all nodes considered in this calculation were not chosen to be reset. From conditional independence, we have

$$\Pr(\cap_{\ell,\ell' \in \mathcal{L}} \psi_{\ell\ell'}|\mathcal{C}(X)) = \prod_{\ell \in \mathcal{L}} \Pr(\cap_{\ell'} \psi_{\ell\ell'}|\mathcal{C}(X)). \tag{17}$$

Since all nodes in a particular group which are also in the same compartment are statistically identical, we may compute the probability

$$\begin{aligned}&\Pr(\cap_{\ell' \in \mathcal{L}} \psi_{\ell\ell'}|\mathcal{C}(X))\\ &= \binom{\mathcal{C}_{g\ell}(X)}{\{\psi_{\ell\ell'}\}_{\ell' \in \mathcal{L}}} \prod_{\psi_{\ell\ell'}>0} \Pr(X_g^\ell \to X_g^{\ell'}|\mathcal{C}(X))^{\psi_{\ell\ell'}}\end{aligned} \tag{18}$$

where $\binom{\mathcal{C}_{g\ell}(X)}{\{\psi_{\ell\ell'}\}}$ is the multinomial coefficient, and we define $\Pr(X_g^\ell \to X_g^{\ell'}|\mathcal{C}(X))$ as the probability of a device in group g and compartment ℓ transitioning to compartment ℓ', given the current compartmental membership count $\mathcal{C}(X)$, i.e.

$$\Pr(X_g^\ell \to X_g^{\ell'}|\mathcal{C}(X)) \triangleq$$
$$\begin{cases}(1-P_{Y_{g0}})\prod_{g' \in \mathcal{G}}(1-P_{Y_{gg'}})^{\mathcal{C}_{g'I}(X)} & \ell = S, \ell' = S\\ 1-(1-P_{Y_{g0}})\prod_{g' \in \mathcal{G}}(1-P_{Y_{gg'}})^{\mathcal{C}_{g'I}(X)} & \ell = S, \ell' = I\\ 1-P_{Z_g^{I\to S}} & \ell = I, \ell' = I\\ P_{Z_g^{I\to S}} & \ell = I, \ell' = S\\ 1-P_{Z_g^{R\to S}} & \ell = R, \ell' = R\\ P_{Z_g^{R\to S}} & \ell = R, \ell' = S.\end{cases} \tag{19}$$

Note that the validity of (19) can be verified by studying (1)-(4), and noting the independence of the random variables $\{Y_{ij}\}$ and $\{Z_i^{\ell\to\ell'}\}$. Note also that the use of the multinomial coefficient in (18) accounts for the number of ways the $\mathcal{C}_{g\ell}(X)$ devices can be assigned to make the required transitions $\{\psi_{\ell\ell'}\}$ occur.

Assembling this calculation, we have

$$\begin{aligned}&\Pr(\mathcal{C}_g(X) \to \mathcal{C}_g(Y)|\mathcal{C}(X), d) =\\ &\sum_{\psi \in \Psi_{g|d}(X,Y)} \prod_{\ell \in \mathcal{L}} \binom{\mathcal{C}_{g\ell}(X)}{\{\psi_{\ell\ell'}\}_{\ell' \in \mathcal{L}}} \prod_{\psi_{\ell\ell'}>0} \Pr(X_g^\ell \to X_g^{\ell'}|\mathcal{C}(X))^{\psi_{\ell\ell'}}.\end{aligned} \tag{20}$$

Since there are three elements in $\mathcal{L}$, $\Psi_{g|d}(X,Y)$ is a subset of $(\Phi_3(n_{dg}))^3$. Hence, Lemma 1 implies that evaluating (20) takes at most $O({n_{dg}}^6)$ time. Computing the transition probabilities for each group individually and then computing their product to obtain the joint probability takes at most $O(n_g {n_d}^6)$ operations, and as there are $O({n_d}^{6n_g})$ entries in the table of transition probabilities, we can compute the entire table in $O(n_g {n_d}^{6n_g+6})$ time, as claimed. ∎

Note that this complexity analysis is quite conservative, as the set $\Psi_{g|d}(X,Y)$ will have less than $O({n_{dg}}^6)$ elements, and there are many transitions which occur with probability zero, and so need not be explicitly computed (e.g., consider any transition in which $d_{gS} + d_{gI} < \mathcal{C}_{gR}(Y) - \mathcal{C}_{gR}(X)$). However, this argument alone is enough to verify that (7) can be represented as a Markov decision process with state and action spaces which grows polynomially with respect to n_d, and this representation can be computed in polynomial time. Since Markov decision processes can be solved in polynomial time (see, e.g., [19, Theorem 1]), we then have that (7) is a polynomial-time problem. This is an important result in principle, as it demonstrates that under broad assumptions, optimal reset policies can be computed without explicitly using the state space representation $\mathcal{X}$, or action set representation $\mathcal{A}$, and so may be computed efficiently.

IV. An Example Application

We study an example in this section. We consider a case where the network consists of $n_d = 20$ devices, all of one type. Time advances at six increments per minute, i.e. every time step is ten seconds in duration. Infected devices spread malware to unaffected devices with probability 0.5 at each time step, i.e. $P_{Y_{ij}} = 0.5$ for all $(i,j) \in [n_d]^2$. The attacker infects an uninfected node with probability 0.05 at each time, i.e. $P_{Y_{i0}} = 0.05$. This corresponds to the attacker re-installing malware on vulnerable devices once approximately every three minutes. We set $P_{Z_i^{R\to S}} = 0.167$, which corresponds to a reset taking approximately one minute. As a utility function, we take

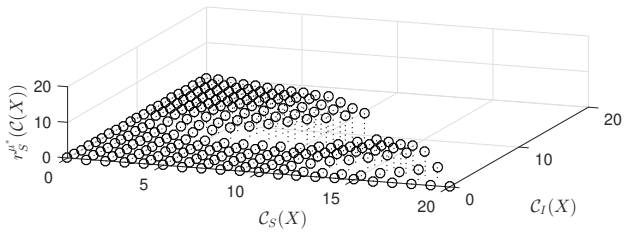
$$\mathcal{U}(X) = \mathbb{I}_{\{\sum_{i \in [n_d]} X_i^I \leq 10\}} \sum_{i \in [n_d]} X_i^S + 0.95 X_i^I.$$

This function captures a scenario in which infected devices can still perform their required task when a malware attack is happening, but at a slightly reduced capacity. However, if too many devices become infected (here, more than 10), some critical system component becomes inoperable. This is precisely what happens when DDoS attacks are launched using security cameras: the quality of the camera's surveillance capability declines negligibly, but after sufficiently many become infected, they can launch a sufficiently strong attack to shut down a web server [20]. To finish specifying the problem, we set the discount factor $\lambda = 0.99$.

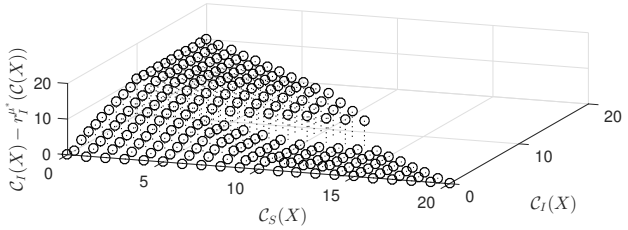
Note that the space $\mathcal{X}$ contains $3^{20} \approxeq 3.5 \times 10^9$ elements, while the space $\mathcal{A}$ contains $2^{20} \approxeq 1 \times 10^6$ elements. As such, directly solving (7) would be prohibitively expensive. In the lumped representation developed in Section III, $\mathcal{C}(X)$ takes only 231 distinct values, and the action set $\mathcal{D}$ contains only 231 distinct actions. In principle, our model reduction technique enables the specified problem to be solved efficiently.

We use the computation outlined by (20) to compute the transition probabilities for the lumped problem, and use value iteration to solve (13) (see [18, Chapter 1] for relevant background, and [21] for a freely available software package which can perform the relevant computations). Figure 1 provides a depiction of the optimal control policy computed. Studying this figure reveals that the structure of the computed optimal policy is complicated. There are regions of the state space in which susceptible devices are reset preemptively, and regions in which several infected devices are not reset.

It is not clear that such a policy can be anticipated intuitively. However, it is worth investigating the performance of a heuristic policy, to determine if we have gained

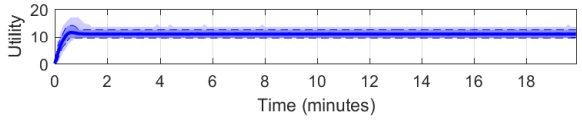


(a) A depiction of the structure of an optimal policy of the example studied in Section IV. The axes marked $\mathcal{C}_S(X)$ and $\mathcal{C}_I(X)$ denote the count of susceptible and infected nodes in a particular state. The vertical axis denotes the number of susceptible nodes forced to initiate a reset under the optimal policy.

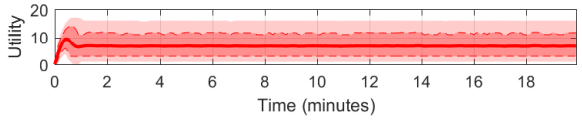


(b) A depiction of the structure of an optimial policy of the example studied in Section IV. The axes marked $\mathcal{C}_S(X)$ and $\mathcal{C}_I(X)$ denote the count of susceptible and infected nodes in a particular state. The vertical axis denotes the number of infected nodes less the number of infected nodes forced to initiate a reset under the optimal policy.

Fig. 1: Optimal reset policy of the problem studied in Section IV.



(a) Simulation of Optimal Reset Policy



(b) Simulation of Heuristic Reset Policy

Fig. 2: A comparison of the performance of the process under the optimal control policy $\mu^\star$ (Figure 2a) and a policy which immediately resets all infected nodes and never resets any susceptible devices (Figure 2b). The dark lines are the sample expectation, the dark shaded and light shaded regions contain the middle 80% and 98% of the sample trajectories, respectively.

substantial performance by computing an optimal policy. Figure 2 studies this, where the performance of the heuristic policy in which all infected devices are forced to reset, and no susceptible devices are forced to reset. Ten thousand sample trajectories were generated. It can be readily seen that the optimal policy significantly outperforms the heuristic policy.

V. Conclusions and Future Work

We have studied the design of optimal reset strategies for protecting networked systems against malware attacks. We have demonstrated that optimal reset strategies may be efficiently computed, provided the network to be managed is comprised of a small number of groups of identical devices.

Directly, the methods discussed here can be applied to managing networks to be both secure and efficient. Indirectly, our technical approach applies more broadly. We hope the discussion provided here inspires epidemic control researchers to incorporate similar thoughts into their own work.

References

[1] R. Hallman, J. Bryan, G. Palavicini, J. Divita, and J. Romero-Mariona, "IoDDoS — The Internet of Distributed Denial of Sevice Attacks - A Case Study of the Mirai Malware and IoT-Based Botnets," *Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security*, no. April, pp. 47–58, 2017.

[2] G. Kambourakis, C. Kolias, and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," *Proceedings - IEEE Military Communications Conference MILCOM*, vol. 2017-Octob, pp. 267–272, 2017.

[3] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[4] R. Anderson and R. Anderson, "Why Information Security is Hard," *Annual Computer Security Applications Conference*, 2001.

[5] M. H. R. Khouzani, S. Sarkar, and E. Altman, "Dispatch then Stop : Optimal Dissemination of Security Patches in Mobile Wireless Networks," in *Proceedings of the 49th IEEE Conference on Decision and Control*, (Atlanta, GA), pp. 2354–2359, 2010.

[6] M. H. R. Khouzani, S. Sarkar, and E. Altman, "Optimal Control of Epidemic Evolution," in *IEEE INFOCOM*, no. i, (Shanghai, China), 2011.

[7] S. Eshghi, M. H. Khouzani, S. Sarkar, and S. S. Venkatesh, "Optimal Patching in Clustered Malware Epidemics," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 283–298, 2016.

[8] S. Eshghi, S. Sarkar, and S. S. Venkatesh, "Visibility-Aware Optimal Contagion of Malware Epidemics," *IEEE Transactions on Automatic Control*, vol. 62, no. October, pp. 5205–5212, 2017.

[9] L. Sun, W. Huang, P. S. Yu, and W. Chen, "Multi-Round Influence Maximization (Extended Version)," 2018.

[10] L. Bilge and T. Dumitras, "Before We Knew It: an Empirical Study of Zero-Day Attacks in the Real World," *Proceedings of the 2012 ACM Conference on Computer and Communications Security – CCS'12*, pp. 833–844, 2012.

[11] S. Egelman, C. Herley, and P. C. van Oorschot, "Markets for zero-day exploits," *Proceedings of the 2013 workshop on New security paradigms workshop - NSPW '13*, pp. 41–46, 2013.

[12] Z. Shan, I. Neamtiu, Z. Qian, and D. Torrieri, "Proactive restart as cyber maneuver for Android," *Proceedings - IEEE Military Communications Conference MILCOM*, vol. 2015-Decem, pp. 19–24, 2015.

[13] B. Thompson, J. Morris-King, and H. Cam, "Effectiveness of proactive reset for mitigating impact of stealthy attacks on networks of autonomous systems," *2016 IEEE Conference on Communications and Network Security, CNS 2016*, pp. 437–441, 2017.

[14] S. Gómez, A. Arenas, J. Borge-Holthoefer, S. Meloni, and Y. Moreno, "Discrete-time Markov chain approach to contact-based disease spreading in complex networks," *EPL*, vol. 89, 2010.

[15] N. A. Ruhi and B. Hassibi, "SIRS epidemics on complex networks: Concurrence of exact Markov chain and approximated models," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 2919–2926, 2015.

[16] N. J. Watkins and G. J. Pappas, "Control of Generalized Discrete-time SIS Epidemics via Submodular Function Minimization," *IEEE Control Systems Letters*, 2018.

[17] W. B. Powell, *Approximate Dynamic Programming*, vol. 2. 2011.

[18] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, Massachusetts: Athena Scientific, third ed., 2007.

[19] C. H. Papadimitriou and J. N. Tsitsiklis, "The Complexity of Markov Decision Processes," *Mathematics of Operations Research*, vol. 12, no. 3, pp. 441–450, 1987.

[20] U. Lindqvist and P. G. Neumann, "The future of the internet of things," *Communications of the ACM*, vol. 60, no. 2, pp. 26–30, 2017.

[21] I. Chadès, G. Chapron, M.-J. Cros, F. Garcia, and R. Sabbadin, "MDPtoolbox: a multi-platform toolbox to solve stochastic dynamic programming problems," *Ecography*, vol. 37, pp. 916–920, 2014.