SAND2016-7055C

State Space Representations for Dynamic Modeling of Moving Target Defense Systems

Jason R. Hamlet
Sandia National Laboratories
Albuquerque, NM
jrhamle@sandia.gov

Christopher C. Lamb
Sandia National Laboratories
Albuquerque, NM
cclamb@sandia.gov

ABSTRACT

Moving Target Defense is an emerging security paradigm in which systems dynamically mutate in order to shift the system attack surface and frustrate would-be attackers. The dynamic nature of some Moving Target Defenses opens the possibility of modeling them with dynamic systems approaches, such as state space representations familiar from control and systems theory. In this paper, we present state space models for Moving Target Defenses, provide an analysis of their properties, and suggest approaches for using them.

Keywords

Moving Target Defense; modeling; attacker-defender interactions

1. INTRODUCTION

Moving Target Defense (MTD) approaches allow systems to dynamically update and adjust to their operating environment proactively or in response to perceived or detected threats or attacks. The intention is to provide legitimate system users and defenders with an advantage over attackers, since those attackers will find it more difficult to determine the current status of a system, or may not be able to rely on the system being in the same configuration when they launch their attack as it was when they performed reconnaissance, developed the attack, or otherwise performed their planning and preparation activities. Of course, MTDs can also make it more difficult for legitimate system users, defenders, and administrators to know the current configuration of their systems, which potentially creates opportunities for attackers. In this work, we consider state space models for studying the dynamic interactions between attackers and defenders with MTDs at their disposal.

2. RELATED WORK

Many MTDs have been suggested in the literature. A survey of them has been provided by Okhravi *et al.* [8]. The attacker-defender dynamics in MTD have previously been studied using game theoretic models. Colbaugh and Glass analyze MTD strategies against adaptive adversaries under the assumption that the available defensive systems are independent so that attacks are effective against a single system [4]. Carter allows attacks to be effective against some subset of systems, and further assumes that the defender does not know which systems are vulnerable and is not able to detect exploits [2]. Van Dijk introduces the FlipIt game to model advanced persistent threats and targeted attacks in which one player has complete control of a resource, but in which it is not known which player controls the resource until a player makes a move, which comes at some cost [12]. Jones extends FlipIt to MTD by allowing defenders to "morph" the system to disrupt attacker knowledge [6]. Prakesh modifies FlipIt by allowing attackers to detect when control of the resource is gained by the defender, and to allow multiple target resources with objectives over the number controlled [10]. Miehling studies MTDs using partially observable Markov decision processes [7].

State space representations have been used to study both linear and nonlinear systems for many decades. They have found particular success in control engineering, where they are used to model physical systems by inputs, outputs, and state variables, and provide a convenient means of studying the time-evolution of such systems. State space models are useful in control engineering for understanding the influence of inputs and disturbances on system state even though the models are simplified approximations of the underlying physical systems.

The paper is organized as follows. In Section 3 we introduce our state space approach for modeling MTDs, and present several state space representations and an analysis of each. Section 4 describes how our representations might be used in practice, and concluding remarks and suggestions for further research are provided in Section 5.

3. APPROACH

We begin the modeling process by making several assumptions. We assume that the attacker and the defender can each influence the system state by applying inputs to the system. These inputs may be corrupted by noise. We further assume that attackers and defenders can at least partially observe the system state. These output observations may also be corrupted by noise. Additionally, we assume that there is no guarantee that either party can impact all of the state variables, or that their observations of the state are correct.

Third ACM Workshop on Moving Target Defense 2016 Vienna, Austria
© 2016 ACM. ISBN 123-4567-24-567/08/06...$15.00
DOI: 10.475/123_4

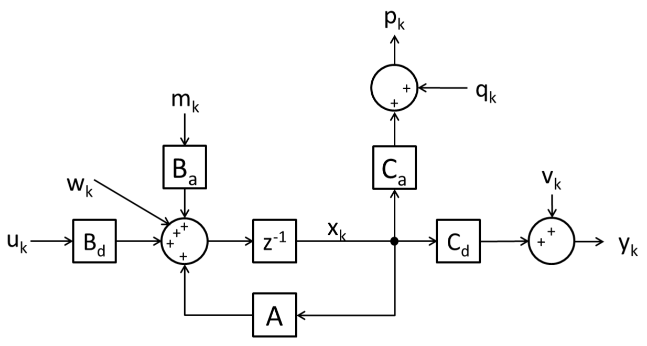

Figure 1: Open-loop state space representation for moving target defense

3.1 Open Loop

Figure 1 shows the simplest open-loop state space representation that conforms to these assumptions. It is a mixed deterministic-stochastic system approach for modeling the dynamic behavior of MTD systems. By inspecting Figure 1 we see that

$$\mathbf{x_{k+1}} = A\mathbf{x_k} + B_d\mathbf{u_k} + B_a\mathbf{m_k} + \mathbf{w_k} \tag{1}$$

$$\mathbf{y_k} = C_d\mathbf{x_k} + \mathbf{v_k} \tag{2}$$

$$\mathbf{z_k} = C_a\mathbf{x_k} + \mathbf{q_k} \tag{3}$$

where $\mathbf{x_k} \in \mathbb{R}^n$ is the state vector, $\mathbf{y_k} \in \mathbb{R}^{q_d}$ and $\mathbf{p_k} \in \mathbb{R}^{q_a}$ are the output vectors as observed by the defender and attacker, respectively, $\mathbf{u_k} \in \mathbb{R}^{p_d}$ and $\mathbf{m_k} \in \mathbb{R}^{p_a}$ are the defender and attacker's respective input vectors, $\mathbf{w_k}$ is a stochastic disturbance input, $\mathbf{v_k}$ and $\mathbf{q_k}$ are random vectors impacting the defender's and attacker's respective observations of the state, $A \in \mathbb{R}^{n\times n}$ is the system matrix, $B_d \in \mathbb{R}^{n\times p_d}$ and $B_a \in \mathbb{R}^{n\times p_a}$ are the input matrices for the defender and attacker, respectively, and $C_d \in \mathbb{R}^{q_d\times n}$ and $C_a \in \mathbb{R}^{q_a\times n}$ are the defender's and attacker's respective output matrices.

To represent an MTD with this model we must determine how to construct each of these matrices and vectors. We have previously developed a dependency graph based approach for analyzing cyber defenses [5]. In that approach different classes of actors, such as system users, administrators, and attackers, are each assumed to have some set of dependencies that they must fulfill in order to accomplish their goal. Some of these may be shared between the different classes of actors, while others will not be. Each of the dependencies has some costs associated with fulfilling it. We interpret the state of the system at time k as a vector of costs for satisfying each of the n edges in the dependency graph. It is then natural that the system matrix, A, be an adjacency matrix representation of the dependency graph. Consequently, $a_{ij} = 1$ if there is an edge from node i to node j in the dependency graph, and $A_{ij} = 0$ otherwise. The $n \times p_d$ matrix B_d maps the defender's control input $\mathbf{u_k}$. One option is for the input $\mathbf{u_k}$ to define the change in each cost metric for each edge in the dependency graph as a result of some defender action. Consequently, we will have $p_d > n$. In this interpretation B_d will be a block diagonal matrix with its entries averaging the impact of the metrics to obtain an overall change in cost for an edge. For example, if there are six metrics and we use an unweighted average to combine them into a cost, then B_d will have the form

$$\begin{bmatrix} \boldsymbol{\alpha} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\alpha} & \mathbf{0} & \dots & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\alpha} & \dots & \mathbf{0} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{\alpha} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix}$$

where

$$\boldsymbol{\alpha} = \begin{bmatrix} 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \end{bmatrix}$$

$$\mathbf{0} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The attacker's input matrix B_a and input vector $\mathbf{m_k}$ are defined similarly, but will have nonzero entries along the entire diagonal. Alternatively, we can have $B_d \in \mathbb{R}^{n\times n}$ and $B_a \in \mathbb{R}^{n\times n}$, which will cause the input matrices to impact each metric equally. In this formulation, the input vectors $\mathbf{u_k}$ and $\mathbf{m_k}$ represent the change in cost for fulfilling a dependency. This form is more amenable to analysis of closed-loop feedback systems.

We interpret $\mathbf{y_k} = C_d\mathbf{x_k} + \mathbf{v_k}$ as the defender's view of the current system state $\mathbf{x_k}$. Consequently, each of the q_d rows of C_d will have a single 1 in a column corresponding to one of the dependency graph's edges. If $q_d < n$ then some of the edges are not visible to the defender. Oftentimes this will be the case, since we do not necessarily assume that the defender has knowledge of all of the attacker's dependencies. Similarly, the attacker's view of the system state is $\mathbf{z_k} = C_a\mathbf{x_k} + \mathbf{q_k}$ and so C_a also has a single 1 in each of its q_a rows. In general, we can have $q_a < n$ to indicate that the attacker has incomplete knowledge of the dependency graph. However, in accordance with Shannon's maxim that "the enemy knows the system" [11] we will usually assume that $q_a = n$ and that $C_a = I$, providing the attacker with visibility into each of the graph's edges. Note, however, that while the attacker may have knowledge of each of the edges, the random vector $\mathbf{q_k}$ may prevent the attacker from having perfect knowledge of $\mathbf{x_k}$.

We say that the system is *defender observable* if

$$\mathbb{O}_d = \begin{bmatrix} C_d & C_dA & C_dA^2 & \dots & C_dA^{n-1} \end{bmatrix}^T \tag{4}$$

has rank n and that it is *defender controllable* if

$$\mathbb{C}_d = \begin{bmatrix} B_d & AB_d & A^2B_d & \dots & A^{n-1}B_d \end{bmatrix} \tag{5}$$

has rank n. Similarly, the system is *attacker observable* if

$$\mathbb{O}_a = \begin{bmatrix} C_a & CaA & C_aA^2 & \dots & C_aA^{n-1} \end{bmatrix}^T \tag{6}$$

has rank n and it is *attacker controllable* if

$$\mathbb{C}_a = \begin{bmatrix} B_a & AB_a & A^2B_a & \dots & A^{n-1}B_a \end{bmatrix} \tag{7}$$

has rank n.

Now, consider the defender's view of the system's transfer function. Taking Z-transforms, we obtain

$$zX(z) - zx_0 = AX(z) + B_dU(z) + B_aM(z) + W(z)$$
$$Y(z) = C_dX(z) + V(z)$$

by rearranging the state equation and substituting into the expression for $Y(z)$ we obtain

$$\begin{aligned} Y(z) &= C_d(zI - A)^{-1}zx_0 \\ &+ C_d(zI - A)^{-1}\left[B_dU(z) + B_aM(z) + W(z)\right] + V(z) \end{aligned} \tag{8}$$

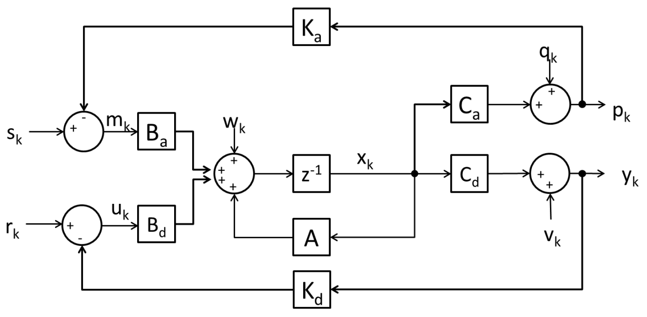

Figure 2: Output feedback state space representation for moving target defense.

The transfer function observed from the defender's perspective is $H_d(z) = Y(z)/U(z)$ and so

$$H_d(z) = C_d(zI - A)^{-1}\left[B_d + \frac{B_aM(z) + W(z)}{U(z)}\right] + V(z)/U(z) \tag{9}$$

Ideally for the defender the $B_aM(z)$ term will be null, indicating no attacker activity. The defender can estimate the value of this term from output measurements by solving eqn. 8 for $B_aM(z)$, which yields

$$B_aM(z) = C_d^{-1}(zI - A)[Y(z) - V(z)] - B_dU(z) - W(z) \tag{10}$$

Similarly, the transfer function from the attacker's perspective is $H_a(z) = P(z)/M(z)$. Manipulating the equations produces the attacker's view of the output as

$$P(z) = C_a(zI - A)^{-1}[B_dU(z) + B_aM(z) + W(z)] + Q(z) \tag{11}$$

and

$$H_a(z) = C_a(zI - A)^{-1}\left[B_a + \frac{B_dU(z) + W(z)}{M(z)}\right] + Q(z)/M(z) \tag{12}$$

The attacker can estimate the defender's influence on the system by solving eqn. 11 for $B_dU(z)$ to obtain

$$B_dU(z) = C_a^{-1}(zI - A)[P(z) - Q(z)] - B_aM(z) - W(z) \tag{13}$$

From both the attacker's and the defender's perspective, this open loop system is stable if the eigenvalues of A fall within the unit circle.

3.2 Output Feedback

Now, we add output feedback and set points to our representation, as shown in Figure 2. This allows the defender and the attacker to attempt to drive the system state, as they observe it, to some desired value. State feedback is not an option for defenders since in general the state is not observable to the defender, and in practice may not be visible to the attacker, either. Considering Figure 2 we first note that

$$\mathbf{x_{k+1}} = A\mathbf{x_k} + B_d\mathbf{u_k} + B_a\mathbf{m_k} + \mathbf{w_k} \tag{14}$$
$$\mathbf{y_k} = C_d\mathbf{x_k} + \mathbf{v_k} \tag{15}$$
$$\mathbf{z_k} = C_a\mathbf{x_k} + \mathbf{q_k} \tag{16}$$

where

$$\mathbf{u_k} = \mathbf{r_k} - K_d\mathbf{y_k} \tag{17}$$
$$\mathbf{m_k} = \mathbf{s_k} - K_a\mathbf{p_k} \tag{18}$$

and where we have

$K_d \in \mathbb{R}^{p_d \times q_d}$ is the defender's feedback gain

$K_a \in \mathbb{R}^{p_a \times q_a}$ is the attacker's feedback gain

and where $\mathbf{r_k}$ and $\mathbf{s_k}$ are the defender and attacker set points, respectively. We begin by finding the defender and attacker transfer functions. Considering the defender first, we assume zero initial conditions and take z-transforms of the state and output

$$zX(z) = AX(z) + B_dU(z) + B_aM(z) + W(z)$$
$$U(z) = R(z) - K_d[C_dX(z) + V(z)]$$
$$Y(z) = C_dX(z) + V(z)$$

substituting $U(z)$ into the first expression and expanding and rearranging terms yields

$$X(z) = (zI - A + B_dK_dC_d)^{-1} \times [B_dR(z) + B_aM(z) - B_dK_dV(z) + W(z)] \tag{19}$$

Substituting this into the expression for $Y(z)$ produces

$$Y(z) = C_d(zI - A + B_dK_dC_d)^{-1} \times [B_dR(z) + B_aM(z) - B_dK_dV(z) + W(z)] + V(z) \tag{20}$$

and so we obtain the closed loop transfer function seen by the defender as

$$H_{cld}(z) = \frac{Y(z)}{R(z)} = \tag{21}$$

$$C_d(zI - A + B_dK_dC_d)^{-1}\left[B_d + \frac{B_aM(z) - B_dK_dV(z) + W(z)}{R(z)}\right] + \frac{V(z)}{R(z)}$$

Now, in the defender's ideal case the $B_aM(z)$ term is the null vector. The defender can use a measurement of $Y(z)$ to estimate the value of $B_aM(Z)$. Solving for $B_aM(Z)$ in eqn. 20 gives

$$B_aM(z) = C_d^{-1}(zI - A + B_dK_dC_d)[Y(z) - V(z)] - B_dR(z) + B_dK_dV(z) - W(z) \tag{22}$$

The defender may be able to use this estimate of the attacker's influence on the system to counteract the attacker's actions.

We may also desire to express the transfer function from R to Y while incorporating the attacker's input vector $\mathbf{m_k}$. If we assume knowledge of the attacker's behavior and design a control system that incorporates it, we may then be able to study how closely we can approximate this situation when the defender either does not have knowledge of the attacker's system, or when the defender can only estimate the attacker's influence, for instance, with eqn. 22. We begin by considering the system state and both defender and attacker inputs

$$zX(z) = AX(z) + B_dU(z) + B_aM(z) + W(z)$$
$$U(z) = R(z) - K_d[C_dX(z) + V(z)]$$

$$M(z) = S(z) - K_a\left[C_a X(z) + Q(z)\right]$$

Substituting the expressions for $U(z)$ and $M(Z)$ into the expression for $X(Z)$ we obtain

$$X(z) = \left(zI - A + B_d K_d C_d + B_a K_a C_a\right)^{-1} \times \\ \left[B_d R(z) + B_a S(z) - B_d K_d V(z) - B_a K_a Q(z) + W(z)\right] \quad (23)$$

Now, since we have that $Y(z) = C_d X(z) + V(z)$ then the transfer function from R to Y is

$$\frac{Y(z)}{R(z)} = C_d\left(zI - A + B_d K_d C_d + B_a K_a C_a\right)^{-1} \times \\ \left[B_d + \frac{B_a S(z) - B_d K_d V(z) - B_a K_a Q(z) + W(z)}{R(z)}\right] + \frac{V(z)}{R(z)} \quad (24)$$

which is the same as eqn. 21 when B_a is the zero matrix. Finally, we also note from eqn. 23 that the system is stable if the roots of $zI - A + B_d K_d C_d + B_a K_a C_a$ are within the unit circle.

We can determine the attacker's transfer function $Y_{cla} = \frac{P(z)}{S(z)}$ by taking z-transforms of the state and output, substituting the expression for $M(z)$ into the expression for $X(z)$, expand and rearrange terms and substitute the result into the expression for $P(z)$ to obtain

$$P(z) = C_a\left(zI - A + B_a K_a C_a\right)^{-1} \times \\ \left[B_d U(z) + B_a S_z - B_a K_a Q(z) + W(z)\right] + Q(z) \quad (25)$$

and so the attacker's closed loop transfer function is

$$H_{cla}(z) = \frac{P(z)}{S(z)} = \quad (26)$$

$$C_a\left(zI - A + B_a K_a C_a\right)^{-1}\left[B_a + \frac{B_d U(z) - B_a K_a Q(z) + W(z)}{S(z)}\right] + \\ \frac{Q(z)}{S(z)}$$

The attacker can use eqn. 25 to estimate the defender's influence on the system, $B_d U(z)$, by calculating

$$B_d U(z) = C_a^{-1}\left(zI - A + B_a K_a C_a\right)\left[P(z) - Q(z)\right] - B_a S(z) + \\ B_a K_a Q(z) - W(z)$$

Now, let's consider the controllability and observability of this system. We begin by writing

$$\mathbf{u_k} = \mathbf{r_k} - K_d \mathbf{y_k} = \mathbf{r_k} - K_d\left(C_d \mathbf{x_k} + \mathbf{v_k}\right) \quad (27)$$

$$\mathbf{m_k} = \mathbf{s_k} - K_a \mathbf{p_k} = \mathbf{s_k} - K_a\left(C_a \mathbf{x_k} + \mathbf{q_k}\right) \quad (28)$$

considering the defender first, we substitute this expression for $\mathbf{u_k}$ into the expression for $\mathbf{x_{k+1}}$ to obtain

$$\mathbf{x_{k+1}} = A\mathbf{x_k} + \left(B_d \mathbf{r_k} - B_d K_d C_d \mathbf{x_k} - B_d K_d \mathbf{v_k}\right) + B_a \mathbf{m_k} + \\ \mathbf{w_k}$$

grouping terms provides

$$\left(A - B_d K_d C_d\right)\mathbf{x_k} + B_d\left(\mathbf{r_k} - K_d \mathbf{v_k}\right) + B_a \mathbf{m_k} + \mathbf{w_k}$$

and so the defender's observability and controllability matrices are

$$\mathbb{O}_d = \left[\begin{array}{cccc} C_d & C_d\left(A - B_d K_d C_d\right) & \dots & C_d\left(A - B_d K_d C_d\right)^{n-1} \end{array}\right]^T \quad (29)$$

$$\mathbb{C}_d = \left[\begin{array}{cccc} B_d & \left(A - B_d K_d C_d\right) B_d & \dots & \left(A - B_d K_d C_d\right)^{n-1} B_d \end{array}\right] \quad (30)$$

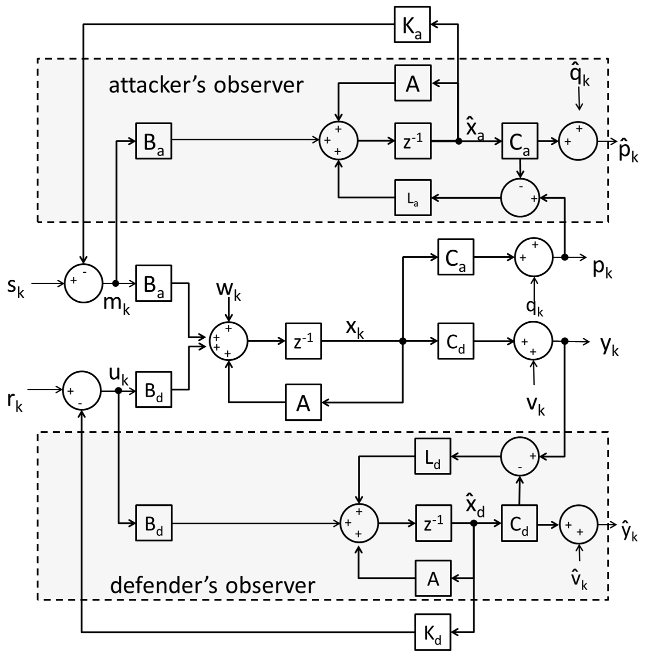

Figure 3: State feedback with state observer state space representation for moving target defense

Considering the attacker's perspective, we substitute the expression for $\mathbf{m_k}$ into the expression for $\mathbf{x_{k+1}}$ from which the attacker's observability and controllability matrices are

$$\mathbb{O}_a = \left[\begin{array}{cccc} C_a & Ca\left(A - B_a K_a C_a\right) & \dots & C_a\left(A - B_a K_a C_a\right)^{n-1} \end{array}\right]^T \quad (31)$$

$$\mathbb{C}_a = \left[\begin{array}{cccc} B_a & \left(A - B_a K_a C_a\right) B_a & \dots & \left(A - B_a K_a C_a\right)^{n-1} B_a \end{array}\right] \quad (32)$$

Here, the system is defender stable if the eigenvalues of $A - B_d K_d C_d$ are within the unit circle. The system is attacker stable if the eigenvalues of $A - B_a K_a C_a$ are within the unit circle.

3.3 State Feedback

To overcome the defender's inability to measure the current state, we can also consider state observer feedback representations, as shown in 3. Before beginning, we note that the defender may not be aware of all of the attacker's edges, and that this prevents the defender from having an observable system. Due to this, if the defender constructs a state observer feedback system then information about unobservable modes will not be included in the output prediction error term, and the defender's state observer will only be stable if all of the unobservable modes are stable [9].

Considering Fig. 3, we first write down the state equations as

$$\mathbf{x_{k+1}} = A\mathbf{x_k} - D_d K_d \widehat{\mathbf{x}}_{\mathbf{d,k}} + B_d \mathbf{r_k} - B_a K_a \widehat{\mathbf{x}}_{\mathbf{a,k}} + B_a \mathbf{s_k} + \mathbf{w_k} \tag{33}$$

$$\widehat{\mathbf{x}}_{\mathbf{d,k+1}} = A\widehat{\mathbf{x}}_{\mathbf{d,k}} + B_d \mathbf{u_k} + L_d\left(\mathbf{y_k} - C_d \widehat{\mathbf{x}}_{\mathbf{d}}\right) \tag{34}$$

$$\widehat{\mathbf{x}}_{\mathbf{a,k+1}} = A\widehat{\mathbf{x}}_{\mathbf{a,k}} + B_a \mathbf{m_k} + L_a\left(\mathbf{p_k} - C_a \widehat{\mathbf{x}}_{\mathbf{a}}\right) \tag{35}$$

$$\mathbf{y_k} = C_d \mathbf{x_k} + \mathbf{v_k} \tag{36}$$

$$\mathbf{p_k} = C_a \mathbf{x_k} + \mathbf{q_k} \tag{37}$$

$$\mathbf{u_k} = -K_d \widehat{\mathbf{x}}_{\mathbf{d}} + \mathbf{r_k} \tag{38}$$

$$\mathbf{m_k} = -K_a \widehat{\mathbf{x}}_{\mathbf{a}} + \mathbf{s_k} \tag{39}$$

where $\widehat{\mathbf{x}}_{\mathbf{d,k}}$ and $\widehat{\mathbf{x}}_{\mathbf{a,k}}$ are the defender's and the attacker's estimates of the state at time k, respectively and

$L_d \in \mathbb{R}^{n\times q_d}$ is the defender's observer gain

$L_a \in \mathbb{R}^{n\times q_a}$ is the attacker's observer gain

We begin by finding the transfer function from the defender's perspective. First, we take the z-transform of eqn. 33 and solve for $X(z)$, to obtain

$$X(z) = \left(zI - A + B_d K_d\right)^{-1} \times \left[B_d K_d E_d(z) + B_d R(z) + W(z) + B_a M(z)\right] \tag{40}$$

Substituting this into the z-transform of eqn. 36 and divide by $R(z)$ to find

$$\frac{Y(z)}{R(z)} = C_d\left(zI - A + B_d K_d\right)^{-1} \times \left[B_d + \frac{B_d K_d E_d(z) + W(z) + B_a M(z)}{R(z)}\right] + \frac{V(z)}{R(z)} \tag{41}$$

From the defender's perspective, this system is stable if the roots of $zI - A + B_d K_d$ are within the unit circle. The defender's observability and controllability matrices are

$$\mathbb{O}_d = \left[\begin{array}{cccc} C_d & C_d\left(A - B_d K_d\right) & \ldots & C_d\left(A - B_d K_d\right)^{n-1}\end{array}\right]^T \tag{42}$$

$$\mathbb{C}_d = \left[\begin{array}{cccc} B_d & \left(A - B_d K_d\right) B_d & \ldots & \left(A - B_d K_d\right)^{n-1} B_d\end{array}\right] \tag{43}$$

By substituting eqn. 40 into eqn 36 and solving for $B_a M(z)$ we can obtain the defender's estimate of the attacker's influence on the system as

$$B_a M(z) = C_d^{-1}\left(zI - A + B_d K_d\right) \times \left[Y(z) - V(z)\right] - B_d K_d E_d(z) - B_d R(z) - W(z) \tag{44}$$

Similarly, we can obtain the transfer function from the attacker's perspective as

$$\frac{P(z)}{S(z)} = C_a\left(zI - A + B_a K_a\right)^{-1} \times \left[B_a + \frac{B_a K_a E_a(z) + W(z) + B_d U(z)}{S(z)}\right] + \frac{Q(z)}{S(z)} \tag{45}$$

From the attacker's perspective the system is stable if the roots of $zI - A + B_a K_a$ are within the unit circle, and the attacker's observability and controllability matrices are

$$\mathbb{O}_a = \left[\begin{array}{cccc} C_a & C_d\left(A - B_a K_a\right) & \ldots & C_a\left(A - B_a K_a\right)^{n-1}\end{array}\right]^T \tag{46}$$

$$\mathbb{C}_a = \left[\begin{array}{cccc} B_a & \left(A - B_a K_a\right) B_a & \ldots & \left(A - B_a K_a\right)^{n-1} B_a\end{array}\right] \tag{47}$$

The attacker can use measurements of $P(z)$ to estimate the defender's influence by

$$B_d U(z) = C_a^{-1}\left(zI - A + B_a K_a\right) \times \left[P(z) - Q(z)\right] - B_a K_a E_a(z) - B_a S(z) - W(z) \tag{48}$$

Now, let's consider the situation in which the defender has knowledge of the attacker's use of state-observer feedback. If we consider the error terms

$$\mathbf{e_{d,k}} = \mathbf{x_k} - \widehat{\mathbf{x}}_{\mathbf{d,k}}$$

$$\mathbf{e_{a,k}} = \mathbf{x_k} - \widehat{\mathbf{x}}_{\mathbf{a,k}}$$

then we can write the state estimates as

$$\widehat{\mathbf{x}}_{\mathbf{d,k}} = \mathbf{x_k} - \mathbf{e_{d,k}} \tag{49}$$

$$\widehat{\mathbf{x}}_{\mathbf{a,k}} = \mathbf{x_k} - \mathbf{e_{a,k}} \tag{50}$$

Substituting eqns. 49 and 50 into eqn. 33 produces

$$\mathbf{x_{k+1}} = \left(A - B_d K_d\right)\mathbf{x_k} + B_d K_d \mathbf{e_{d,k}} - B_a K_a \mathbf{x_k} + B_a K_a \mathbf{e_{a,k}} + B_d \mathbf{r_k} + B_a \mathbf{s_k} + \mathbf{w_k} \tag{51}$$

Now, by taking the z-transform of eqn. 51 and substituting into the z-transform of eqn. 36 we obtain the defender's transfer function

$$\frac{Y(z)}{R(z)} = C_d\left(zI - A + B_d K_d + B_a K_a\right)^{-1} \times \left[B_d + \frac{B_a S(z) + B_d K_d E_d(z) + B_a K_a E_a(z) + W(z)}{R(z)}\right] + \frac{V(z)}{R(z)} \tag{52}$$

The defender can obtain an estimate of the attacker's influence by substituting z-transforms of eqns. 39 and 50 into eqn. 44, solving for the B_a terms, and substituting $X(z) = C_d^{-1}\left(Y(z) - V(z)\right)$ which yields

$$B_a\left[S(z) + K_a E_a(z) - K_a C_d^{-1}\left(Y(z) - V(z)\right)\right] = C_d^{-1}\left(zI - A + B_d K_d\right) \times \left[Y(z) - V(z)\right] - B_d K_d E_d(z) - B_d R(z) - W(z) \tag{53}$$

4. OUTPUT FEEDBACK EXAMPLE: APPLICATION TO NETWORK RANDOMIZATION APPROACHES

Now, we apply the output feedback modeling approach to studying network randomization techniques. In particular, we consider IP and port hopping and route randomization. We use the output feedback model to study the dynamics of a system, as viewed by defenders and attackers of that system, in response to defender and attacker actions on the system. This study allows us to explore the effects that attackers and defenders can have on the system and the

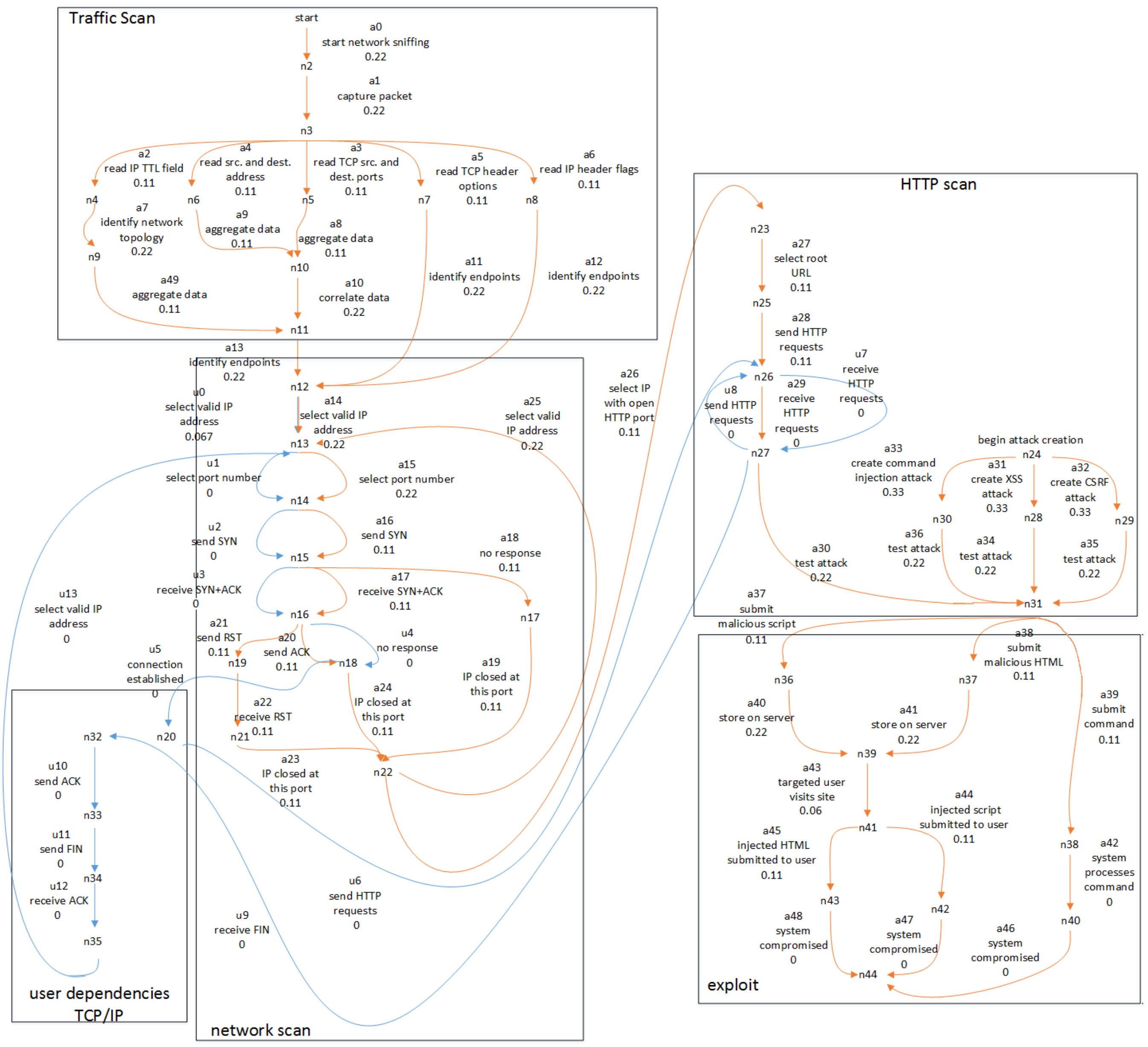

Figure 4: Network randomization dependency graph showing costs to fulfill dependencies prior to applying network randomization. The user's costs are all 0 because our user costs metrics consider percent increase in costs.

dynamics of various defender strategies for deploying the mitigations.

First, we must combine the dependency graphs of the three defenses into a single dependency graph. Then, we take the adjacency matrix representation of this graph as the system matrix, A. As in section 3.1, the B_d and B_a matrices map the defender's and attacker's inputs, which represent the impact of the inputs on each of the metrics affiliated with each of the edges in the dependency graph, to their effect on the system state, x_k, which represents the cost of fulfilling the dependency represented by an edge in the graph. These input matrices define how the attacker and defender actions can influence the system state. Consequently, it is through the B_d matrix that we define the impact of a moving target defense on the system, In the output feedback system, we additionally have attacker and defender feedback gain matrices K_d and K_a. These matrices determine how much influence the defender's and attacker's measurement of the system state have on the selection of the next defender or attacker action.

The defender's input set point, r_k and attacker's input set point, s_k, represent the desired values of each of the metrics for each of the edges from the defender and attacker perspectives. While more nuanced set points are possible, we generally assume that the defender will want to minimize all of the costs for fulfilling all of the defender's dependencies while also maximizing all of the costs for fulfilling each of the attacker's dependencies. The attacker will desire to minimize each of the attacker's costs. While it is unlikely that either the defender or the attacker will be able to reach these ideal set points, the defender can use the distance from this set point, as measured by u_k, to help in selection of an appropriate moving target defense. That is, we envision a playbook, consisting of a set of j matrices $\{B_{d0} \dots B_{dj}\}$ where j is the number of defensive moves available to the defender. Each of these defenses will impact the system by changing some of the cost metrics for some of the edges in the dependency graph. At each time step k the defender can choose which defense, if any, to apply to the system. If no defensive move is made then the defender simply selects the initial input matrix B_{d0}, which represents the cost of fulfilling dependencies with no defenses in place, and has no impact on the system at that time step. Considering this, we have time-varying input matrices $B_d(k)$ and $B_a(k)$, and our problem is to identify which of the available defender moves $\{B_{d0} \dots B_{dj}\}$ to select at each time step. We call a sequence of moves $(B_d(0) \dots B_d(k))$ a *defender strategy*. There are many ways to undertake the analysis. We can define a strategy and then study how it performs by solving the state equations to find the resulting system dynamics. We can also use the defender's knowledge of the system at a given time k, as provided by the output vector y_k, to choose the next defense $B_{d,k+1}$ according to some rulebook and study the resulting evolution of the system dynamics. The resulting sequence of input matrices is a defender strategy guided by the rulebook, and so its performance can be used to help us define acceptable rulebooks and strategies for various attacker scenarios. Similar comments and definitions hold for the attacker.

Figure 4 shows an example dependency graph for standard TCP/IP and HTTP network communications on which we will study network randomization approaches [3]. The costs for fulfilling dependencies in this graph are the initial costs without any network randomizations in place. We begin by briefly describing the dependencies. First, the attacker performs a traffic scan to identify endpoints and valid IP addresses, a network scan to identify IP addresses with open HTTP ports, and finally an HTTP scan. The attacker also creates and tests attacks, and then launches an exploit by submitting a malicious script, HTML, or command to the server and injecting it to the targeted user when the user visits the site. This compromises the user's system. The user has dependencies related to communicating with HTTP over TCP/IP. The network randomization approaches impact the user and attacker dependencies in various ways. IP randomization impacts the attacker's ability to identify the network topology and correlate traffic scan data to identify valid endpoints. This makes it more difficult to select a valid IP address, which in turn increases the difficulty of finding an open HTTP port over which to submit a malicious payload to a targeted user. It also increases the user's cost for select valid IP addresses, communicating over TCP/IP, and sending HTTP requests. The impact on the user is primarily from increased network latency, and from small increases in CPU and memory requirements. Port randomization also complicates identification of the network topology, increases the difficulty of selecting valid HTTP ports, and consequently of submitting malicious payloads to the user. It increases the user's costs for selecting valid ports, communicating over TCP/IP, and sending HTTP requests by increasing network latency and slightly impacting CPU and memory requirements. Path randomization primarily impacts the attacker's ability to correlate traffic scan data for network mapping. It increases the user's costs for communicating over TCP/IP and sending HTTP requests by increasing network latency. When we combine these mitigations we impact the union of the dependencies impacted by the individual dependencies, although the change in costs for fulfilling the dependencies is usually less than the sum of the change for the individual mitigations.

We study this system with our output feedback state space model. For brevity, we consider only the influence of defensive actions through the B_d matrices and the defender's set point $\mathbf{r_k}$, although similar assignments hold for the attacker's input matrices B_a and set point $\mathbf{s_k}$.

We begin by transforming our dependency graph model of the system depicted in 5 into our output feedback state space representation. In this representation the system matrix $A \in \mathbb{R}^{n \times n}$ where $n = 44$ represents the 44 nodes in our system and the edges connecting these nodes. We also have input matrices $B_d, B_a \in \mathbb{R}^{44 \times 44}$. The system's initial conditions represent the cost for fulfilling each of the dependencies without any defenses in use. For this example, we consider IP randomization. This defense impacts the defender by slightly increasing the memory and processing requirements for selecting valid IP addresses at $u0$ and $u13$, while also slightly degrading the system and network stability for doing so. It also increases latency for standard TCP/IP communications at $u2 - u4$, $u6$, $u8$, and $u10 - 12$. From the attacker's perspective the time and cost for acquiring access and knowledge to identify the network topology ($a7$), correlate traffic scan data ($a10$), and select valid IP addresses ($a14$) is increased, as is the the unpredictability and frequency of defense movement at these edges. The unpredictability, frequency of movement, and time and cost for acquiring the necessary knowledge to select an IP address with open HTTP port($a26$), submit a malicious script, HTTP, or command ($a37 - 39$), and identify when the targeted user visits the compromised web site ($a43$) are also increased. We represent these changed costs for fulfilling dependencies by creating a new input matrix B_{d1} with diagonal entries reflecting the new overall costs for fulfilling each dependency. Note that since several edges can originate at the same node, as with node $n3$, the influence of a defense on a particular node is represented by adding the influence on all of the edges originating at that node. This is a consequence of our adjacency matrix representation of the graph, which causes the state space models to view the system from the perspective of costs for reaching a node, rather than the costs for traversing edges. This aliases the influence of edges originating from a single node into a single entry of the Bd or B_a matrices, and also into a single entry of the set point, state, and output vectors. The defender wants the cost of fulfilling every attacker edge to reach its maximum value of 1 and for the cost of fulfilling each defender edge to reach its minimum value of 0. These desires are codified in the set point vector $\mathbf{r_k}$ by setting the i^{th} element of $\mathbf{r_k}$ to γ where γ is the number of attacker edges originating at node i. We also have output matrices $C_d, C_a \in \mathbb{R}^{n \times n}, n = 44$. We generally assume that the attacker has full knowledge of the dependency graph, and so $C_a = I_n$. In this case, we also assume that the defender is aware of all of the nodes in the dependency graph, and so we also have $C_d = I_n$. However, this will not always be the case. In particular, if we want to model an attack that the defender is not familiar with then we would include nodes and edges in the dependency graph that are unknown to the defender, and the diagonal elements of C_d that correspond to these nodes would be set to 0.

With these assignments in place we can begin to study the system. First, we consider properties of the open-loop system with B_{d0} in place, indicating that none of the network randomization options is applied. First, we find that the system is defender observable, but that it has 14 states that are uncontrollable by the defender. A pole-zero plot reveals that the open loop system is unstable, but we also find that all of the uncontrollable modes are located at the origin, and so we know that the system can be stabilized by the feedback controller. We use the Linear Quadratic Regulator (LQR) design technique to find a feedback gain matrix $K_d \in \mathbb{R}^{n \times n}, n = 44$ that stabilizes the closed-loop system [1]. The feedback has no impact on the observability or controllability of the system. We next find the time-domain response of the open and closed loop system and compare them. Results for the attacker edge $(n25, n26)$ are presented in figure 5. In figure 5a we see the impact of IP randomization on attacker dependency $A28 = (n28, n26)$, which is not directly influenced by the IP randomization. Here, we see that the IP randomization moves the the attacker's costs closer to the set point than the system with no network randomization. Similar behavior is observed in figure 5b, which shows the dynamics of attacker dependencies $A25$ and $A26$. Here, edge $A26$ is directly influenced by the IP randomization but $A25$ is not. As before, the system that includes IP randomization is closer to obtaining the set point. Notice that, since two attacker edges originate at node $n22$ the set point in figure 5b is 2. While we do not show them here, many of the attacker and user dependencies do not exhibit any dynamic behavior under the baseline or IP randomiza-

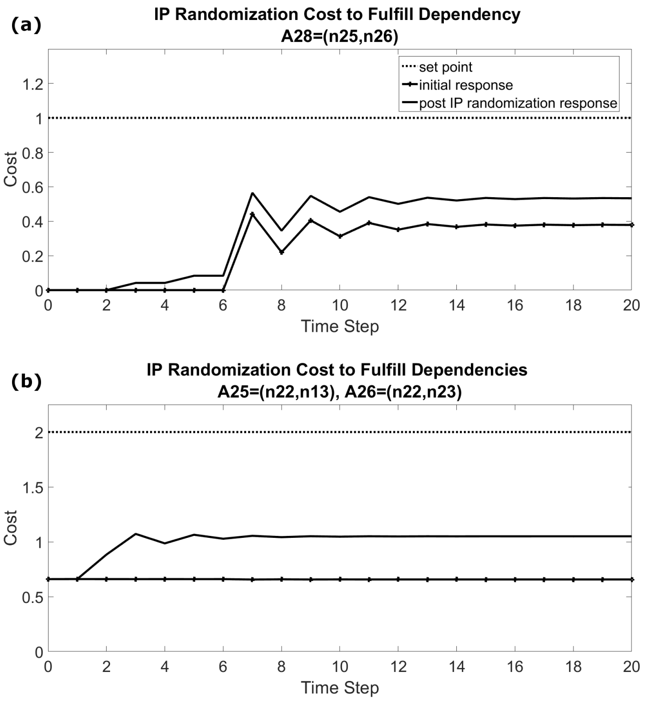

Figure 5: (a) Response for adversary dependencies $A28 = (n28, n26)$ and (b) $A25 = (n22, n13)$ and $A26 = (n22, n23)$ before and after IP randomization. Edges $A25$ and $A28$ are not directly impacted by the IP randomization.

tion conditions. Some of these dependencies are uncontrollable and some are not influenced by these particular inputs. This will usually be the case. If the defender wishes to impact all of the attacker's dependencies, then the defender must identify a set of defenses that collectively cover all of the attacker's edges. Oftentimes, this will not be possible, and so we can instead use graph analysis techniques such as graph centrality and community detection to identify those attacker edges that are most beneficial to target. Further discussion on this topic appears in [5].

5. CONCLUSION AND FUTURE WORK

We have introduced several state space representations for modeling the dynamic interactions between attackers and defenders in cyber systems. These modeling approaches may be particularly well suited to analyzing moving target defense systems since in these systems the defensive posture changes over time. We focused our attention on describing several potential models and developing the mathematical fundamentals for analyzing them, including various transfer functions, observability, and controllability. Future work can use this foundation to explore mode advanced concepts by adopting concepts from the wealth of existing control theory work to these new models.

Future work should also study the resiliency of systems to perturbations and modeling inaccuracies. In particular, it will be important to understand how sensitive the defender is to inaccuracies in modeling the attacker. For example, sensitivity to an attacker that uses a different feedback structure than that modeled by the defender, or an attacker that uses an open-loop strategy in which attacks are not influenced by observation of the system behavior are both of interest. It will also be important to study sensitivity to inaccuracies in the dependency graph. These may be due to unknown and un-modeled dependencies or inaccurately modeled dependencies. Additionally, our network randomization example only considers the defender's influence on the system. Future analysis should study the system dynamics that result from a combination of defender and attacker inputs to the system.

6. ACKNOWLEDGMENTS

This work is being performed under the Laboratory Directed Research and Development Program at Sandia National Laboratories. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

7. REFERENCES

[1] P. J. Antsaklis and A. N. Michel. *Linear systems.* Springer Science & Business Media, 2006.

[2] K. M. Carter, J. F. Riordan, and H. Okhravi. A game theoretic approach to strategy determination for dynamic platform defenses. In *Proceedings of the First ACM Workshop on Moving Target Defense*, pages 21–30. ACM, 2014.

[3] A. Chavez, W. Stout, and S. Peisert. Techniques for the dynamic randomization of network attributes. In *Proceedings of the 49th Annual International Carnahan Conference on Security Technology*, 2015.

[4] R. Colbaugh and K. Glass. Predictability-oriented defense against adaptive adversaries. In *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*, pages 2721–2727. IEEE, 2012.

[5] J. Hamlet and C. Lamb. Dependency graph analysis and moving target defense selection. In *submitted*, 2016.

[6] S. Jones, A. Outkin, J. Gearhart, J. Hobbs, J. Siirola, C. Phillips, S. Verzi, D. Tauritz, S. Mulder, and A. Naugle. Evaluating moving target defense with pladd. Technical report, Sandia National Laboratories (SNL-NM), Albuquerque, NM (United States), 2015.

[7] E. Miehling, M. Rasouli, and D. Teneketzis. Optimal defense policies for partially observable spreading processes on bayesian attack graphs. In *Proceedings of the Second ACM Workshop on Moving Target Defense*, pages 67–76. ACM, 2015.

[8] H. Okhravi, T. Hobson, D. Bigelow, and W. Streilein. Finding focus in the blur of moving-target techniques. *Security & Privacy, IEEE*, 12(2):16–26, 2014.

[9] A. V. Oppenheim and G. C. Verghese. *Signals, systems and inference.* Prentice Hall, 2015.

[10] A. Prakash and M. P. Wellman. Empirical game-theoretic analysis for moving target defense. In *Proceedings of the Second ACM Workshop on Moving Target Defense*, pages 57–65. ACM, 2015.

[11] C. E. Shannon. Communication theory of secrecy systems. *Bell system technical journal*, 28(4):656–715, 1949.

[12] M. Van Dijk, A. Juels, A. Oprea, and R. L. Rivest. Flipit: The game of "stealthy takeover". *Journal of Cryptology*, 26(4):655–713, 2013.