A Game-Theoretic Two-Stage Stochastic Programing Model to Protect CPS against Attacks

Clara Novoa†, Khan Siddique†, Mina Guirguis* and Alireza Tahsini*
†Ingram School of Engineering, Industrial Engineering, Texas State University
*Department of Computer Science, Texas State University

***Abstract*—Securing Cyber-Physical Systems (CPS) against cyber-attacks is a challenging problem as it requires continuously checking control and measurement signals for errors at runtime. The assignment of check blocks to check those signals must be performed using the available but possibly incomplete or uncertain information and within the delay bounds dictated by the control loop. Moreover, the assignment should remain unpredictable against an adversary who is able to observe/probe the assignment placed and adapt her attack methods accordingly. Due to the large number of potential check blocks that can be assigned, their varying effectiveness in detecting a wide range of cyber-attacks, and the uncertainty on the exact number of signals that need to be checked and protected, finding such strategic assignment is a critical endeavor. This paper presents two-stage stochastic programming models for equipping the CPS control loops with the proper check blocks to secure them. The formulation is based on a game theoretical approach to enable the defender to find an optimal randomized (i.e., mixed strategy) assignment of check blocks while abiding to the control-loop constraints. The models incorporate uncertainty in the number of signals to be checked/protected and capture various degrees of overhead in the operation of the check blocks. We illustrate the superiority of our results, in terms of the value of the stochastic solution, when compared to other assignment strategies. We validate our results through a Simulink-based model for a component in the operation of an autonomous vehicle.**

I. Introduction

The integration of Cyber-Physical Systems (CPS) into transportation, healthcare, power, manufacturing and industrial systems presents unprecedented challenges in ensuring their safety, security and trustworthinesses. Cyber-attacks on such systems can have severe consequences as their impact would cross from the cyber realm into the physical world – evident from previous incidents such as the recent attacks on the power-grid in Ukraine. As the integration of CPS continues, we expect cyber-attacks to take on new roles that go beyond what we experience today in terms of Denial of Service (DoS), ransomware and data breaches. Soon, there will be new attacks that can cause vehicles to veer off the road in autonomous driving settings, manipulate devices responsible for power generation and consumption, and exploit robotics/drone/industrial systems for malicious and terrorism-related activities.

One important aspect in securing CPS is ensuring that the right control and measurement signals are propagated within the control loop. Recent research efforts have proposed various check blocks to check the signals against spoofing and false data injection attacks [10], [11], [7], [18], [20], [6]. The smart grid, in particular, has received a lot of attention for ensuring correct state estimation and operation through checking signals [6], [7], [18]. Despite such efforts, two major lacking aspects emerge in checking/protecting signals: (1) There appears to be no rigorous approach to properly integrate various check blocks in a meaningful manner for developing an advanced defense system. Most of the work done developed specific check blocks that detect specific attacks. Moreover, these blocks operate in isolation of each other. (2) the operation of check blocks relies on a certain level of certainty in terms of the number of signals that will be checked. Such certainty may be violated during the operation of the CPS. For example, a check block that takes a time t to check a signal may cause severe stability degradation when faced with a surge of signals during the normal, albeit transient, operation of the system.

In [8], the authors present a game theoretical approach to enable the defender to find an optimal randomized (i.e., mixed strategy) assignment of check blocks while abiding to the control-loop constraints. This paper extends this line of work through developing two-stage stochastic programming models that incorporate uncertainty in the number of signals to be checked/protected and capture various degrees of overhead in the operation of the check blocks.

Contributions: We make the following contributions: (1) develop two-stage stochastic programming models that incorporate uncertainty in the number of signals to be checked at runtime. They determine optimal recourse actions after the stochastic parameter(s) have been realized. (2) capture non-linear overhead as the numbers of signals vary during transient operations of the CPS. (3) assess the value of the stochastic approach through extensive simulations experiments and by developing a simulink model of a real CPS system. We demonstrate the effectiveness of the proposed approach when compared to other assignment strategies. To the best of our knowledge, this work is the first to investigate the use of two-stage stochastic programming models to assign check blocks in CPS settings within a game-theoretic formulation.

Paper organization: Section II presents related works on game-theoretic techniques to secure CPS emphasizing how this work is different. The proposed two-stage stochastic programming models are presented in Section III. The numerical results from solving the stochastic models is in Section IV. The validation of the model with stochastic number of signals through a simulation in Simulink is in Section V and the paper conclusion and further developments are in Section VI.

II. Related Work

Game theory has been utilized in a wide range of studies to analyze the security of CPS and networked-control systems [5], [13]. Researchers model and analyze security issues as security games in which attacker and defender have conflicting objectives – the attacker tries to maximize the damage to the system while the defender tries to minimize the loss. Through such models, secure control laws can be obtained as well as defense mechanisms that detect cyber-attacks.

In [7], the authors represent a human-CPS interaction model in a smart city, where the CPS elements are considered to form a graph. In that graph, the cyber and physical elements are interdependent nodes; consequently, each node is a battlefield in the proposed Colonel Blotto game, where the attacker and defender assign recourses to win a battle. Although that formulation is a very general one, it does not explicitly take into account the specific interaction of resources in each battlefield. The authors in [9] investigate the process of transmitting data from a sensor to a controller in a CPS, while the attacker tries to strike this communication with a jamming attack. By considering power constraints for both players, a static and a dynamic game is formulated for finding the optimal transmission and attack strategies for the players, given a time horizon. Another effort in [12] presents a model for assuring the security in a CPS by solving a parametric game matrix. Those specified parameters express the cost, loss and benefit of players after taking an action. The game is played in each state of a small state space, which captures the dynamic behavior of the system in a normal situation and under attacks. However, computing values of those parameters remained abstract. The work in [22] presents a receding-horizon methodology to approximate an infinite-horizon dynamic Stackelberg game to securely control a CPS. The model assumes two attackers: one can attack the measurement signals and the other one can attack the control signals. The model is a two-level Stackelberg game, where the measurement jammer makes a decision before the operator and the control jammer. In the first-level the measurement jammer is the leader and the operator (controller) and the control jammer are the followers. In the second one, the operator is the leader and the control jammer is the follower. The attackers try to manipulate the signals to destabilize the system, while minimizing the cost of the attack by solving a quadratic program. On the other hand, the operator tries to find a control law that maintains the stability of the system. Our work is different from the above works as it considers a different formulation in which the defender is seeking an assignment of concrete stateless check blocks – that have varying effectiveness to protect against various attack methods while abiding to the delay constraints imposed by the control loop as well as accounting for the uncertainty in the number of signals received. Our work utilizes various defense mechanisms that have been proposed to detect cyber-attacks in CPS [10], [11], [19], [18], [21], [3] and provides a proper framework for integrating those mechanisms in the control loop. In this paper, we utilize the Stackelberg formulation that resembles those models that screen for threats in security games (e.g., [4], [16], [14]). Our model is different, however, as it focuses on the stochastic nature of the signals. Previous evidence of the application of two-stage stochastic programming to cyber-security is scarce. The authors in [1] developed a stochastic network interdiction model based on a probabilistic attack graph with uncertain attack success probabilities on the arcs. They formulated it as a two-stage stochastic mixed-integer linear program. They employed the sample average approximation scheme along with the Bender's decomposition approach to solve the resulting problem. Their model provides an optimal recommendation for countermeasure deployment in a stochastic environment. The authors in [17] model enterprise networks using attack graphs that let to visualize the current and future security state of the network and the optimal steps to protect it from external threads. This work however is outside the scope of CPS.

III. Mathematical Programming Models

A. A Game-Theoretic Framework

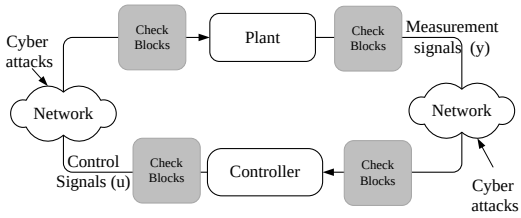

Fig. 1: A general block diagram for a CPS.

We consider a general CPS model composed of a plant and a controller. We let x_k denote the state of the system at time k. A vector of measurement signals y_k is generated from various sensors attached to the plant and is fed to the controller. The controller, in return, generates a vector of control signals, u_k that change the state of the plant. We assume that each signal is transmitted independently from the others and that measurement and/or control signals traverse network segments that are subjected to various types of cyber-attacks (e.g., spoofing, injection, denial of service, etc.). Figure 1 shows a block diagram of the CPS.

We consider a game-theoretic formulation of a 2-player zero-sum game between a defender and an adversary. The defender seeks to protect the CPS through enabling a set of check blocks within the control loop as shown in Figure 1. The adversary seeks to attack the CPS through selecting a target signal (i.e., a measurement or a control signal) to attack and a particular attack method (e.g., spoof, jam, inject, deny). Due to the cost incurred in checking signals, the defender seeks an assignment of check blocks per target signal that does not violate the delay constraints for the correct operation of the CPS while maximizing his utility subject to the adversary choosing her best response. In our model we have:

- **Target Signals** T: Each target signal t corresponds to an actual type of measurement or control signal propagated

within the control loop. The defender gains a utility U_t^p when t is protected and a utility U_t^u when t is unprotected.

- **Attacks** A: The adversary can choose an attack method a (e.g., spoof, jam, inject, deny) from a set of available attacks A on target signal t. We let c denote the attack category which is the tuple $<t, a>$.
- **Check Blocks** B**:** The defender can choose to protect targets from attacks by assigning check blocks within the control loop. We let E_b^a denote the probability of block b in protecting against attack a, and T_b and T_{2b} denote times for executing check block b on an arriving signal. We explain the difference between T_b and T_{2b} right after the model is presented in the next subsection.
- **Number of Signals** N_{te}**:** For each target signal t, we assume that the exact number of signals propagated in the time horizon [0,TH] is unknown, but is dictated based on a set of operational scenarios E. We let N_{te} denotes the number of signals arriving to target t under forecasted scenario e that occurs with probability p_e.
- **Adversary Type** Θ**:** There can be different types of adversaries that value targets differently based on the importance of each signal. The parameter z_θ represents the probability of encountering an adversary of type θ.
- **Delay Capacity** C_t**:** It represents the maximum allowable delay for checking a target signal t by all assigned blocks that does not degrade the operation of the system.

In [8], a Stackelberg game is formulated based on the parameters above being all known with certainty (i.e. deterministic). In [8], the defender commits first to an assignment that seeks to minimize the attacker's best response. The game is solved as a linear optimization problem in which the decision variable is $n_{b,t}$ that denote the assignment of block b to target signal t. In the following section we extend this model to capture uncertainty in the number of signals observed through a two-stage stochastic program.

B. Two-Stage Stochastic Program (Two-SSP) - Stochastic Number of Signals

Stochastic programming is an approach for modeling optimization problems that include parameters that are uncertain, but assumed to lie in a given set of values at the time a decision should be made [15]. In a *Two-SSP*, decisions are divided into two stages. The first one includes decisions done before the values of the random parameter(s) are known, and the second one considers a specific action or recourse to take after the stochastic parameter(s) realize. The recourse decisions look to compensate for any bad effects resulting from the first-stage decisions. *Two-SSP* looks for a policy that is feasible for all (or almost all) the possible parameter realizations by optimizing the expected value of a function that assesses the impacts of first-stage decisions and two-stage recourse actions for each scenario. In a *Two-SSP*, a scenario defines specific values for the possible realization of the model uncertainty.

In our *Two-SSP* model, the defender operates on a time horizon $[0, TH]$. At time zero, he doesn't not know with certainty the total number of signals that will arrive in $[0, TH]$, but he can estimate it based on a given probability distribution. The horizon splits into two stages $[0, T1]$ and $[T1, TH]$. The defender looks to find the assignment of check blocks to signals that maximizes the total expected utility assuming that at time $T1$, when the number of arriving signals is realized in an exact or an almost exact way, the defender implement a recourse action. Such action is to perform an update on the assignments of check blocks to targets.

A standard approach to solve *Two-SSP* is to assume that the random vector of model parameters has a finite number of possible realizations or scenarios (e) with associated probabilities (p_e). Under this assumption, the *Two-SSP* can be formulated as a large linear program known as the extensive form or the *Deterministic Equivalent Model (DEM)*. This work follows this approach; different scenarios e for the number of signals arriving in $[0, TH]$ are considered. The proposed *DEM* that models uncertainty on the number of arriving signals is denoted as *DEM1*. We present the decision variables and the model next.

***DEM1* - Decision Variables**:

- s_θ denote the defender's worst case possible utility given an adversary type θ,
- n_{bt} denote the amount of first-stage assignment of block b to target signal t,
- r_{bte} denote the second-stage recourse decisions. They account for the amount of increase (+) or decrease (-) on the assignment of block b to target t under scenario e that the defender will implement. Thus, r_{bte} can be positive or negative

DEM1 - Mathematical Model

$$\max \qquad S_{DEM1} = \sum_{\theta \in \Theta} z_\theta \mathbb{E}(s_\theta) \tag{1}$$

$$\mathbb{E}(s_\theta) \leq \mathbb{E}(x_c) U_t^p + (1 - \mathbb{E}(x_c)) U_t^u \qquad \forall \theta, c \tag{2}$$

$$\mathbb{E}(x_c) = \max_{b \in B} E_b^a (n_{bt} + \sum_e p_e r_{bte}) \qquad \forall c \tag{3}$$

$$\sum_{b \in B} n_{bt} N_{te} T_b + \sum_{b \in B} |r_{bte}| N_{te} T_{2b} \leq C_t \qquad \forall t, e \tag{4}$$

$$n_{bt} + r_{bte} \leq 1 \qquad \forall b, t, e \tag{5}$$

$$n_{bt} + r_{bte} \geq 0 \qquad \forall b, t, e \tag{6}$$

$$n_{bt} \in \{0, 1\} \qquad \forall b, t \tag{7}$$

$$-k \leq r_{bte} \leq k \qquad \forall b, t, e \tag{8}$$

The objective function given by equation (1) maximizes the expected defender's worst case utility $\mathbb{E}(s_\theta)$, given the probability z_θ of encountering an adversary type θ . Constraint (2) enforces that the expected utility, $\mathbb{E}(s_\theta)$, is the worst possible over all possible attack categories that the adversaries could choose from. Constraint (3) determines the expected probability of thwarting an attack $\mathbb{E}(x_c)$ as the expected probability of thwarting an attack with the *most* effective block. The term inside the sum considers the second-stage assignments of blocks in each scenario with their associated probabilities. Constraint (4) enforces that the time to assign

the first and second-stage blocks to protect target t would not exceed the delay capacity C_t in any scenario e. In (4), T_b is the time to execute the first-stage assignment of block b, and T_{2b} is the additional time to execute the recourse action on block b (i.e. time to increase or decrease the first-stage block assignment in the second-stage). The absolute value of r_{bte} is to consider that all changes to the first-stage block assignment use time and then reduce the available capacity C_t.

Constraints (5) and (6) ensure that the total assignment of any block to any target is in the [0,1] interval in each scenario and that r_{bte} will not exceed n_{bt}. Constraint (7) assures that *DEM1* provides valid fist-stage assignments. Constraint (8) states that due to practical limitations, the recourse decisions, r_{bte}, may be bounded to lie in an interval $[-k,k]$, where k is a model parameter that the defender can vary.

As it occurs in practice, *DEM1* assumes that there is no extra cost for implementing the recourse actions besides the expected increase in delay considered in constraint (4). Thus, the objective function does not have a term to account for the expected cost of the recourse actions. Constraints (3) and (7) make the model a Mixed Integer Program since the solver engine linearizes the *max* function in constraint (3) by adding integer variables. Replacing constraint (7) with $0 \leq n_{bt} \leq 1 \quad \forall b,t$, and keeping the recourse variables continuous the resulting model is notated as *DEM1_relaxed*. This relaxed model obtains marginal or fractional assignments of blocks to targets and the recourse actions to these assignments. In practice, such marginal or fractional assignment can be interpreted as: (1) the blocks are not fully-operating, for instance, watermarking with a lower resolution, testing a limited set of conditions, etc. or (2) only a fraction of the arriving signals to a target is checked, which is implementable through sampling signals based on those marginal values.

C. Two-SSP - Stochastic Number of Signals and Times for Enabling Blocks Modeled with Piecewise Linear Functions

In *DEM1*, times T_b and T_{2b} are constant. This assumption fails if there is an overhead to assign the blocks or if these times depend on the fractions of block assigned. One example of this occurs if assigning machine learning blocks in which the execution time may be non-linear. To extend *DEM1*, T_b is modeled as a piecewise linear function of the fraction of first-stage block assignments n_{bt} and T_{2b} is modeled as a piecewise linear function of the second-stage block assignments r_{bte}. In a general form, the piecewise linear function used for modeling T_b has n slopes, $T_1, T_2, T_3, ..., T_n$, with breakpoints, $x_1, x_2, x_3, ..., x_n$ that are fractional values of n_{bt} and the piecewise linear function for T_{2b} has n slopes with n breakpoints that are fractional values of r_{bte}. *DEM1* with T_b and T_{2b} in (4) modeled as piecewise linear functions is denoted as *DEM2*. The piecewise linear function for T_b is:

$$T_b = \begin{cases} T_1 & \text{if} \quad 0 < n_{bt} \leq x_1 \\ T_2 & \text{if} \quad x_1 < n_{bt} \leq x_2 \\ ... & \\ T_n & \text{if} \quad x_{n-1} < n_{bt} \leq x_n \end{cases}$$

To simplify the notation, in the remainder of this paper, the suffix *relaxed* will be dropped from the model names and *DEM1* and *DEM2* will correspond to the relaxed versions.

IV. Numerical Results

A. Mathematical Solution to $DEM1$ and $DEM2$

Mathematical programming models *DEM1*, given by equations (1)-(8) with constraint (7) relaxed ($0 <= n_{bt} <= 1$), and *DEM2*, given by the same equations but with constraint (4) modified to model T_b and T_{2b} as piecewise linear functions, were coded in the IBM ILOG Optimization Programming Language (OPL). Then the models were mathematically solved with the CPLEX solver under the Integrated Development Environment using the CPS described in subsection B. The algorithm that CPLEX used to solve the mixed integer programs *DEM1* and *DEM2* is Branch and Cut. It runs a Branch and Bound algorithm and uses cutting planes to tighten the linear programming relaxations. Mathematical solutions to *DEM1* and *DEM2* are presented in subsections C and D, respectively.

B. CPS Setup

The CPS used to obtain numerical solutions to *DEM1* and *DEM2* consists of 6 targets, 3 attacks from 1 adversary and 3 check blocks that the defender can enable. The value of k in constraint (8) is set to 0.1, There are three scenarios e, for the number of arriving signals per target (low, medium, high), with probabilities p_e. The transposed matrix for the number of arriving signals per scenario and target N'_{te} and other model parameters are in Table I. Matrix E^a_b represents effectiveness of check blocks to protect against attacks. An effectiveness of 0.8 in entry [1,2] means check block 1 has an 80% probability of detecting attack 2. The first row in the transposed utility matrix, U'_t, shows the defender's negative utility when a target is not defended, the second one shows the defender's no impact in utility if the target is defended. The third and fourth rows show similar information for the attacker's utility.

Parameter	$Value$
N'_{te}	$\begin{bmatrix} 7 & 7 & 7 & 7 & 7 & 7 \\ 10 & 10 & 10 & 10 & 10 & 10 \\ 11 & 11 & 11 & 11 & 11 & 11 \end{bmatrix}$
p_e	$\begin{bmatrix} 0.2 & 0.3 & 0.5 \end{bmatrix}$
T_b	$\begin{bmatrix} 2 & 1 & 1.5 \end{bmatrix}$
T_{2b}	$\begin{bmatrix} 2.4 & 1.2 & 1.8 \end{bmatrix}$
E^a_b	$\begin{bmatrix} 0.95 & 0.8 & 0 \\ 0 & 0 & 0.5 \\ 0 & 0.3 & 0.8 \end{bmatrix}$
C_t	$\begin{bmatrix} 40 & 40 & 40 & 40 & 40 & 40 \end{bmatrix}$
U'_t	$\begin{bmatrix} -200 & -100 & -100 & -100 & -100 & -100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 200 & 100 & 100 & 100 & 100 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

TABLE I: Parameters for the CPS solved

C. Effectiveness of Stochastic Model $DEM1$

The value of the stochastic solution (VSS) for *DEM1* is computed to numerically assess the effectiveness of *DEM1*. The procedure to compute the VSS consists of 4 steps.

It follows the presentation in [2]. *Step1: DEM1* is solved with the CPLEX solver to find the the expected worst case utility S_DEM1, the fist-stage block assignments, n_{bt} and the second-stage recourse actions r_{bte}. Such solution is depicted in the second column of Table II. *Step 2:* A deterministic model *DM* with the number of signals arriving to each target, N'_t, fixed to $[10\ \ 10\ \ 10\ \ 10\ \ 10\ \ 10]$, $e = 1$, $p_e = 1$ and no second-stage recourse actions r_{bte} is solved with the CPLEX solver to find the block assignments. *DM* fixes N_t to the average number of signals, which is 9.9 and it is rounded to 10. In *DM*, the time to enable the blocks T_b is $[2\ \ 1\ \ 1.5]$. We omit the presentation of the mathematical model for *DM* to keep the paper concise. *Step3:* The goodness of the assignments given by *DM* are evaluated in the stochastic setting by plugging them into *DEM1* as first-stage assignments. Then, *DEM1* is solved with CPLEX and with only the second-stage recourse decisions as decision variables. Such model is notated as DM_in_DEM1 and its resulting utility is denoted as $S_{DM\_in\_DEM1}$. The solutions to this model are depicted in the last column of Table II. *Step 4:* VSS is computed as S_{DEM1} - $S_{DM\_in\_DEM1}$ and presented in the last row in Table II. The fairness of this comparison stems from the fact that both models $DEM1$ and DM_in_DEM1 are allowed to find for recourse actions, that may be not necessarily the same, to cope with the uncertainty in the number of signals. The VSS or utility gain from solving the stochastic model *DEM1* instead of the deterministic model *DM* is significant and equal to 52.576.

Further experimentation included to use the CPLEX solver to numerically solve models *DEM1* and *DM_in_DEM1* and compute the VSS for *DEM1* under 9 settings that result from assuming 3 levels for E_b^a, the probability of block b in protecting against attack a (low, medium, high), and 3 levels for U_t, the utility or importance for the targets (low, medium, high). Five runs were done in each setting for a grand total of 45 CPLEX runs for each model. The 5 different values for E_b^a in the low, medium and high setting were randomly selected from the intervals [0, 0.5], [0.5,0.7] and [0.7, 1.0], respectively. The total utility was kept to 700 to do fair comparisons among the studied settings. In the low utility case, all the targets had an utility of 116.67. In the medium case, one of the targets had a higher utility and it was randomly chosen between 200 and 300. The other targets had equal values. In the high utility case, one target had a higher utility randomly chosen between 300 and 450 and the other ones had an equal value. The VSS results under each of the 9 experimental settings were averaged over the 5 runs. These results are presented in Figure 2. In that figure, the inner x-axis labels correspond to the levels for E_b^a, the probability of block b in protecting against attack a, and the outer x-axis labels correspond to the levels for U_t, the utility for the targets. In all the settings studied the gain from solving *DEM1* is evidenced. For all levels of utility of the targets, the VSS or gain from solving the stochastic model *DEM1* is larger as the probability of the blocks to protect against attacks increases and the percentage of increment is almost linear. The increasing trend of VSS suggests the effectiveness of *DEM1* over the deterministic model *DM*.

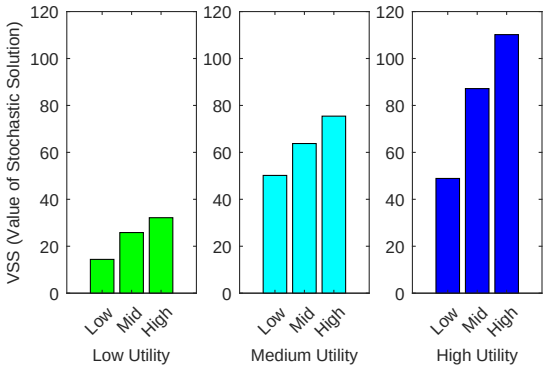

Fig. 2: VSS for *DEM1* for different levels of utility of the targets and effectiveness of the blocks.

D. Results for Stochastic Model $DEM2$

The ranges to generate slopes for the first-stage times, T_b, for the 3 blocks were [0.1, 2.00], [0.05, 1.00], [0.08, 1.50] and the ones for the second-stage times, T_{2b}, were [0.12, 2.40], [0.06, 1.20], [0.09, 1.80]. Nine breakpoints in [0,1] were used. Worst case utility for $DEM2$ resulted equal to the one for $DEM1$ (S_{DEM2} = -40.0) but the first-stage assignment for $DEM2$ slightly reduced as shown in the matrix below. *DEM2* and *DEM1* had similar computational times (80-100 ms).

$$n_{bt}=\begin{bmatrix} 1 & 0.8 & 0.8 & 0.8 & 0.8 & 0.8 \\ 0 & 0 & 0.1 & 0.886 & 0 & 0.1 \\ 1 & 0.85 & 0.85 & 0.7 & 0.75 & 0.8 \end{bmatrix}$$

Additional experiments were performed to observe the effect of the ratio T_{2b}/T_b on the expected worst case utility for *DEM2* if also varying the delay capacity C_t at 3 levels, 10, 20, and 40. Results show that when the second-stage assignment takes less time than the first-stage assignment ($T_{2b}/T_b < 1$), the expected worst case utility is better than when $T_{2b}/T_b > 1$. This trend is more observable if the system delay capacity C_t is low (e.g. 10). This result agrees with the authors' intuition about the behavior of the proposed piecewise functions and their effect on worst case utility.

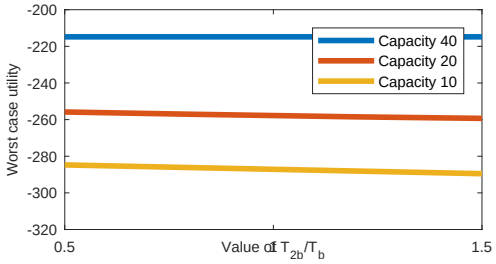

Fig. 3: Effect of the ratio T_{2b}/T_b on worst case utility for *DEM2* at 3 different capacity levels.

Figure 4 shows average expected worst case utilities for *DEM1* and *DEM2* under the same 9 settings described in subsection C. *DEM2* gives a better worst case utility since the piecewise functions have lower times for some amounts of

	DEM1	*DM_in_DEM1*
n_{bt}	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0.086364 & 0 & 0.1 & 0.1 & 0.1 & 0.1 \\ 1 & 0.9 & 1 & 0.81063 & 0.93758 & 0.82 \end{bmatrix}$	$\begin{bmatrix} 0.5714 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0.5714 & 0.625 & 0.625 & 0.625 & 0.625 & 0.625 \end{bmatrix}$
r_{bte}	$\begin{bmatrix} 0 & -0.1 & -0.1 & -0.1 & -0.1 & -0.1 \\ -0.086364 & 0 & -0.1 & -0.1 & -0.1 & -0.1 \\ 0 & -0.1 & -0.1 & -0.1 & -0.1 & -0.1 \end{bmatrix}$ $\begin{bmatrix} 0 & -0.1 & -0.1 & -0.1 & -0.1 & 0 \\ 0 & 0 & -0.1 & -0.1 & 0 & -0.1 \\ 0 & -0.1 & 0 & -0.1 & -0.1 & 0 \end{bmatrix}$ $\begin{bmatrix} 0 & 0 & 0 & -0.1 & 0 & -0.046545 \\ 0.1 & 0 & 0 & -0.1 & -0.1 & -0.1 \\ 0 & 0.1 & 0 & -0.021269 & 0 & -0.1 \end{bmatrix}$	$\begin{bmatrix} 0.1 & -2.9976e-15 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & -0.1 & -0.1 & -0.1 & -0.1 & -0.1 \\ 0.1 & 0.0875 & -0.1 & -0.1 & 0.0875 & -0.1 \end{bmatrix}$ $\begin{bmatrix} 0.1 & 0.093825 & 0.027158 & 0.1 & 0.1 & 0.1 \\ 0.1 & -0.1 & -0.1 & -0.1 & -0.1 & -0.1 \\ 0.1 & -0.1 & -0.1 & 0.025 & -0.1 & -0.1 \end{bmatrix}$ $\begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.056295 & 0.056295 & 0.056295 \\ 0.1 & -0.1 & -0.1 & -0.1 & -0.1 & -0.1 \\ 0.1 & -0.1 & -0.025 & -0.1 & -0.1 & -0.025 \end{bmatrix}$
	$S_{DEM1} = -40$	$S_{DM\_in\_DEM1} = -92.576$
VSS	S_{DEM1} - $S_{DM\_in\_DEM1}$ = 52.576	

TABLE II: Value of the stochastic solution (VSS) for *DEM1*

block assignments if compared to *DEM1* and it favors higher check block assignments for some targets.

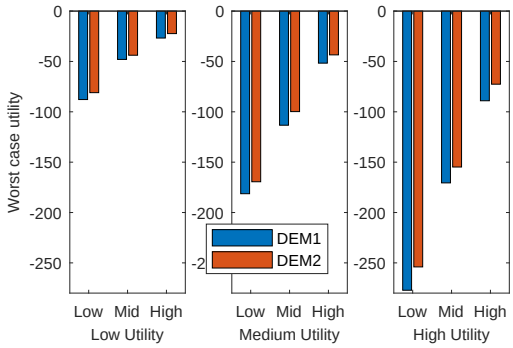

Fig. 4: Expected worst case utility for *DEM1* and *DEM2*

V. Simulink Evaluation

A simulation model of a CPS was built in Matlab Simulink to simulate the solution from *DEM1* and validate the results. Figure 5 represents the simplified diagram of the Simulink model that simulates an simple adaptive cruise control system used in an autonomous driving component to maintain a specified cruise speed and a safe distance from the vehicle in front of it. A signal builder block generates an input signal that causes the lead vehicle to accelerate or break. The velocity of the lead vehicle is regulated by the input signal. The host vehicle has to follow lead vehicle with a preset cruise speed and also maintain a preset minimum distance. If the distance is greater than a set distance, the host vehicle will accelerate until it reaches cruise speed. Then the host vehicle will maintain the cruise speed until the distance from lead vehicle becomes less than the set distance. If the host vehicle reaches the set distance, it will reduce its speed. The input to the host vehicle is also acceleration to modify speed through a PID controller block and a feedback loop.

This proof-of-concept CPS model has four target signals, two measurement signals (i.e. speed of lead vehicle and distance from lead vehicle) and two control signals (i.e. host vehicle acceleration and break). The stochastic number of signals arriving to each target is generated through conditional rate transition blocks. The probability distribution for the number of signals is $[5, 7, 10]$ with probabilities of $[0.2, 0.3, 0.5]$, respectively. The following three types of cyber-attack functions are programmed to manipulate the signal and to prevent the CPS from operating optimally.

- **Minimum Value Attack (MVA):**

$$\bar{y} = \begin{cases} y & \text{if} \quad y > k \\ k & \text{if} \quad y \leq k \end{cases} \tag{9}$$

  MVA fixes the input signal to an arbitrary minimum k. If the value of the signal y is less than k, it denies the signal and sends the minimum value creating a temporary instability in the CPS as for a certain time the system will receive the same value.
- **Offset Attack (OA):**

$$\bar{y} = y + rand[-y, y] \tag{10}$$

  OA adds an arbitrary random number in the range of $[-y, y]$, where y is the value of incoming signal. Then, OA abruptly changes a signal and tends to change the behavior of the CPS.
- **Random Noise Attack (RNA):**

$$\bar{y} = y + N(\mu, \sigma^2) \tag{11}$$

  RNA adds a noise to the original signal that follows a normal distribution with mean μ and variance σ^2. Then RNA can tweak the signal to deceive the check blocks and destabilize the CPS.

The model has three check blocks that can be assigned to the model to detect the attacks and consequently protect the CPS.

- **Parity-based Check Block (PCB):** It marks even and odd signals differently and can detect the attack by reversing the marking procedure. PCB has a high chance of catching a MVA and a fair chance to protect the CPS against RNA and OA.
- **Threshold-based Check Block (TCB):** It compares the incoming signal with a preset value and decides that an attack has happened if the value exceeds the threshold value. TCB is moderately effective against MVA or OA

where the attack function largely manipulates the signal. TCB is not very effective against a RNA.

- **Watermarking Block (WB):** It adds a checksum digit in the least significant digit of the signal. It has very negligible effect on the signal property and behavior. WB is very effective against RNA, but can be less effective against MVA or OA, as random attacks can pick values in a wider range.

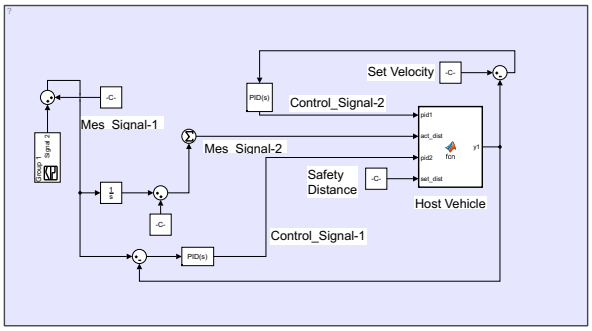

Fig. 5: A simplified block diagram of the Simulink model.

The parts of the simulated system are interconnected through a network that can be subject to the different cyber-attacks. The attack nodes have different attack functions and they activate when the attacker decides to attack. Check blocks are employed before the signals enter the control module or the execution module, thus they can detect whether a signal has been attacked or not. Figure 6 shows the behavior of the simulation model when the system is under attack and when it is under no attack. Under no attack, the model behaves as designed. The host vehicle maintains a preset cruise speed while maintaining a safe distance from lead vehicle. However, when the attack functions are activated the system no longer behaves as designed. The speed of the host vehicle increases beyond the set speed and the distance also goes below the set distance. In some cases, the distance becomes negative and at that point it indicates an accident.

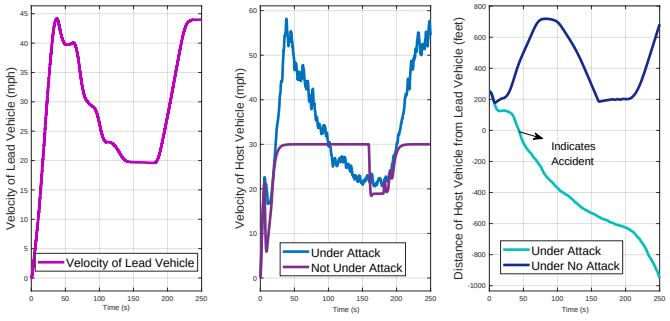

Fig. 6: Behavior of the simulation model under cyber-attacks.

The check block assignments coming from solving the following mathematical models and heuristic methods are plugged into the Simulink model and simulated for 250 seconds: (i) Two-SSP *DEM1* with arriving number of signals to each target following a discrete probability distribution equal to $[5, 7, 10]$ with probabilities $[0.2, 0.3, 0.5]$, (ii) deterministic model *DM* with known number of arriving signals, $N_t = 7$ for each target signal, (iii) a greedy heuristic method that seeks to protect the target with highest utility first using the most effective check block available and then looks to protect the next most valuable target, and (iv) random assignment of check blocks, a method that does not correspond to a well engineered attack but permits to assess the performance of the check block assignments from *DEM1* to a worst case method.

The utility and delay capacity of the targets and the effectiveness of the check blocks are the same in the four compared models/methods. While the simulation was running, the check blocks were active or inactive according to the assignments coming from solving the models/methods. The Simulink model recorded the number of times the check blocks detected an attack. This result let us to compute the percentage of times an attack was detected and thus to estimate the probabilities of detecting an attack as presented in Figure 7. The comparison of the percentage of attacks detected by the different models indicates that Two-SSP *DEM1* gives the best results closely followed by the deterministic model *DM*. The Greedy (G) heuristic seeks to protect the most valuable target signal but can leave other target signals vulnerable and so failing to detect several attacks. The Random (R) method for assigning check blocks has the worst performance.

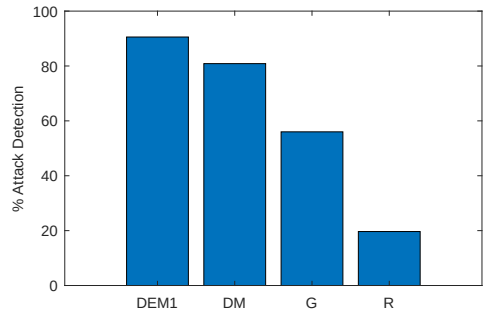

Fig. 7: Estimated probability of detecting an attack.

The estimated probability of thwarting an attack permitted to calculate the worst case utility for each model/method. Figure 8 presents the worst case utility results.

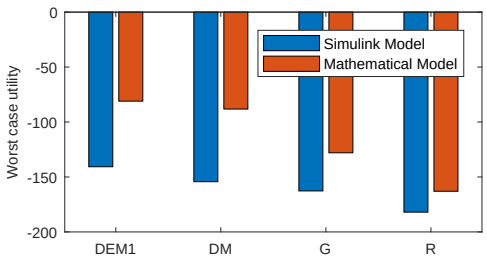

Fig. 8: Worst case utility comparison for simulated methods.

Figure 8 corroborates that in the Simulink model the assignment from *DEM1* gives the "best worst case utility" and

that the assignment from the random (R) approach gives the worst "worst case utility". Figure 8 shows a difference in the worst case utility between Simulink and *DEM1* that partly arises from the estimation of the probability of thwarting an attack done in the Simulink model. Differences in utilities between Simulink and the other methods are because the Simulink model emulates a random number of arriving signals and deterministic (DM), greedy (G) and random (R) methods do not include this behavior. Our study also tested how the methods to assign blocks perform under false attacks. The false positive percentage for different approaches is presented in Fig-9. The Two-SSP *DEM1* gives the least percentage of false positive among the different approaches compared.

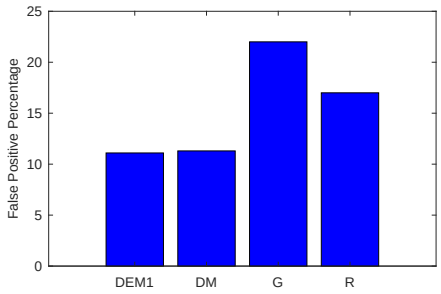

Fig. 9: False positive percentage for the simulated methods.

VI. Conclusions and Further Developments

An important component in securing CPS is ensuring that the correct control and measurement signals are propagated within the CPS control loop. This paper provided a rigorous game-theoretic approach in which check blocks can be integrated efficiently in the control loop. Our approach considers delay constraints of the control loop, effectiveness of the blocks and uncertainty in the number of signals through a two-stage stochastic Stackelberg model. Our solutions are evaluated through extensive numerical analysis and simulink implementation. They demonstrate the value of our stochastic approach when compared to a deterministic mathematical programming model, a greedy heuristic and a random method.

Further developments include to: (1) test the check block assignments given by *DEM1* on a real testbed, (2) assess the model results under multiple type of adversaries, (3) model the probability of check blocks to protect from attacks, E_b^a, as random variables, (4) model and solve the cyber-security problem as a multi-stage mathematical program in which the defender can apply different recourse actions in each stage; such model could represent more closely the recourse options available for continuously checking control and measurement signals in a CPS. Also, study how to integrate the models with algorithms that protect the CPS against coordinated attacks (e.g. replay, covert, false data injection).

References

[1] T. H. Bhuiyan, A. K. Nandi, H. Medal, and M. Halappanavar. Minimizing expected maximum risk from cyber-attacks with probabilistic attack success. In *2016 IEEE Symposium on Technologies for Homeland Security (HST)*, May 2016.

[2] J. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer, New York, NY, 1997.

[3] R. Bobba, K. Rogers, Q. Wang, H. Khurana, K. Nahrstedt, and T. Overbye. Detecting False Data Injection Attacks on DC State Estimation. In *The First Workshop on Secure Control Systems, CPS Week*, 2010.

[4] M. Brown, A. Sinha, A. Schlenker, and M. Tambe. One Size Does Not Fit All: A Game-Theoretic Approach for Dynamically and Effectively Screening for Threats. In *AAAI conference*, 2016.

[5] C. T. Do, N. H. Tran, C. Hong, C. A. Kamhoua, K. A. Kwiat, E. Blasch, S. Ren, N. Pissinou, and S. S. Iyengar. Game theory for cyber security and privacy. *ACM Computing Surveys (CSUR)*, 50(2):30, 2017.

[6] A. Farraj, E. Hammad, A. Al Daoud, and D. Kundur. A game-theoretic analysis of cyber switching attacks and mitigation in smart grid systems. *IEEE Transactions on Smart Grid*, 7(4):1846–1855, 2016.

[7] A. Ferdowsi, W. Saad, B. Maham, and N. B. Mandayam. A colonel blotto game for interdependence-aware cyber-physical systems security in smart cities. In *Proceedings of the 2nd International Workshop on Science of Smart City Operations and Platforms Engineering*, pages 7–12. ACM, 2017.

[8] M. Guirguis, A. Tahsini, K. Siddique, C. Novoa, J. Moore, C. Julien, and N. Dunstatter. Bloc: A game-theoretic approach to orchestrate cps against cyber attacks. In *Proceedings of the IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2018.

[9] Y. Li, L. Shi, P. Cheng, J. Chen, and D. E. Quevedo. Jamming attacks on remote state estimation in cyber-physical systems: A game-theoretic approach. *IEEE Transactions on Automatic Control*, 60(10):2831–2836, 2015.

[10] Y. Liu, P. Ning, and M. Reiter. False Data Injection Attacks against State Estimation in Electric Power Grids. In *the 18th ACM Conference on Computer and Communications Security*, Chicago, IL, November 2009.

[11] Y. Mo and B. Sinopoli. False Data Injection Attacks in Control Systems. In *Proceedings of the 1st workshop on Secure Control Systems*, pages 1–6, 2010.

[12] H. Orojloo and M. A. Azgomi. A game-theoretic approach to model and quantify the security of cyber-physical systems. *Computers in Industry*, 88:44–57, 2017.

[13] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu. A survey of game theory as applied to network security. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10. IEEE, 2010.

[14] A. Schlenker, H. Xu, M. Guirguis, C. Kiekintveld, A. Sinha, M. Tambe, S. Sonya, D. Balderas, and N. Dunstatter. Dont bury your head in warnings: A game-theoretic approach for intelligent allocation of cyber-security alerts. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Melbourne, Australia, 2017.

[15] A. Shapiro and A. Philpott. A tutorial on stochastic programming, 2007.

[16] A. Sinha, T. Nguyen, D. Kar, M. Brown, M. Tambe, and A. Jiang. From Physical Security to Cybersecurity. *Journal of Cybersecurity*, 2015.

[17] A. Subil and S. Nair. Cyber security analytics: A stochastic model for security quantification using absorbing Markov Chains. *Journal of Communications*, 2014.

[18] A. Tiwari, B. Dutertre, D. Jovanović, T. de Candia, P. Lincoln, J. Rushby, D. Sadigh, and S. Seshia. Safety Envelope for Security. In *Proceedings of the 3rd international conference on High confidence networked systems*, pages 85–94. ACM, 2014.

[19] N. Trcka, M. Moulin, S. Bopardikar, and A. Speranzon. A Formal Verification Approach to Revealing Stealth Attacks on Networked Control Systems. In *Proceedings of the 3rd International Conference on High Confidence Networked Systems*, Chicago, IL, April 2014.

[20] D. I. Urbina, J. A. Giraldo, A. A. Cardenas, N. O. Tippenhauer, J. Valente, M. Faisal, J. Ruths, R. Candell, and H. Sandberg. Limiting the impact of stealthy attacks on industrial control systems. In *ACM CCS*, 2016.

[21] Y. Wang, Z. Xu, J. Zhang, L. Xu, H. Wang, and G. Gu. SRID: State Relation Based Intrusion Detection for False Data Injection Attacks in SCADA. In *Computer Security-ESORICS*. Springer, 2014.

[22] M. Zhu and S. Martinez. Stackelberg-game analysis of correlated attacks in cyber-physical systems. In *American Control Conference (ACC), 2011*, pages 4063–4068. IEEE, 2011.