BLOC: A Game-Theoretic Approach to Orchestrate CPS against Cyber Attacks

Mina Guirguis*, Alireza Tahsini*, Khan Siddique†, Clara Novoa†, Justin Moore*,
Christine Julien‡ and Noah Dunstatter*
*Department of Computer Science, Texas State University
†Ingram School of Engineering, Industrial Engineering, Texas State University
‡ Department of Electrical and Computer Engineering, University of Texas at Austin

***Abstract*—Securing Cyber-Physical Systems (CPS) against cyber-attacks is challenging due to the wide range of possible attacks – from stealthy ones that seek to manipulate/drop/delay control and measurement signals to malware that infects host machines that control the physical process. This has prompted the research community to address this problem through developing targeted methods that protect and check the run-time operation of the CPS. Since protecting signals and checking for errors result in performance penalties, they must be performed within the delay bounds dictated by the control loop. Due to the large number of potential checks that can be performed, coupled with various degrees of their effectiveness to detect a wide range of attacks, strategic assignment of these checks in the control loop is a critical endeavor. To that end, this paper presents a coherent runtime framework – which we coin BLOC – for orchestrating the CPS with check blocks to secure them against cyber attacks. BLOC capitalizes on game theoretical techniques to enable the defender to find an optimal randomized use of check blocks to secure the CPS while respecting the control-loop constraints. We develop a Stackelberg game model for stateless blocks and a Markov game model for stateful ones and derive optimal policies that minimize the worst-case damage from rational adversaries. We validate our models through extensive simulations as well as a real implementation for a HVAC system.**

I. Introduction

Motivation: Cyber-Physical Systems (CPS) will be core to most emerging computing systems. A myriad of activities in our lives will rely on the correct operation of these systems, from the transportation and energy domains, to manufacturing and healthcare. Across these domains, ensuring the correctness of our CPS is essential, making them not just *trusted systems* but *trustworthy* ones. Recently, however, new classes of cyber attacks have emerged that demonstrate a true lack of rigorous approaches in ensuring the resiliency of CPS against them especially in highly dynamic and unpredictable environments in which they are required to operate. From the infamous stuxnet worm that targeted the Iranian nuclear plants to the more recent attack on the power grid in Ukraine, cyber attacks on CPS are becoming appealing vehicles for terrorism and terrorism-related activities [6]. Anecdotally, the vast majority of the CPS built are validated through *ad hoc* trial-and-error approaches. Photographs of CPS engineers in hard hats with laptops debugging their CPS attached to buildings and bridges dominate the general opinion of CPS. A recent empirical study of verification and validation techniques used by CPS experts [22] has demonstrated a pervasive frustration with the lack of rigorous tools; confirming that *in situ* trial-and-error debugging is the most commonly used approach.

Despite the recent (and rich) research efforts in identifying attacks on CPS and developing resilient defense mechanisms against them (which we discuss in more details in Section II), a coherent approach in which these mechanisms can be "put" together is lacking. Blindly throwing more checks and defenses at the CPS may not just be unnecessary, but could negatively impact the stability margins of the system allowing more attacks to be mounted. Moreover, once the attacker discovers a particular defense, he can change the attack.

Problem Statement: Ensuring the correctness, timing and integrity of the control and measurement signals requires orchestrating the control loop with various "check blocks" – such as threshold checks, model predictors, learning modules, assertions and action blocks. Due to their different timing, overhead and effectiveness characteristics, it becomes challenging to choose the right one(s), specially against a rational adversary who is *aware of the blocks present and seeks to inflict the maximum damage*. Moreover, they must *abide by the capacity constraints* dictated by the process controlled. In this paper, we present a game-theoretic framework – which we coin BLOC – that dynamically equips the control loop with the appropriate blocks at the right location(s) to protect it against a rational adversary. Through an optimal orchestration of blocks that are the results of mixed strategies, the system can be defended against an adversary that can choose the best attack method against the system. BLOC considers the range of possible attacks, the importance of each measurement and control signal, the nature of each block (e.g., stateless or stateful) as well as the effectiveness and timing of each block.

Contributions: We summarize our contributions below:

1) We present the BLOC framework, which orchestrates the CPS with various check blocks in a coherent manner. These blocks are enabled at four strategic locations with access to the intended control and measurement signals.
2) We present a Stackelberg game formulation between the defender and the adversary. We obtain optimal marginal assignments of *stateless* check blocks that maximize the utility of the defender subject to the adversary's best response.
3) We develop a Markov game model in which the defender uses the maximin value iteration approach to obtain an

optimal policy for assigning check blocks and applies it over a series of sub-games. This model captures the *statefulness* of the check blocks assigned.

4) We assess and validate our models through extensive simulations as well as through a real orchestrated CPS composed of a domestic Heating, Ventilation and Air Conditioning (HVAC) system.

Paper organization: In Section II we discuss related work. In Section III we present our proposed BLOC framework, and Sections IV-A and IV-B present our game-theoretic models for stateless and stateful blocks, respectively. We present our numerical and implementation results in Section V and conclude the paper in Section VI.

II. Related Work

There has been a lot of research efforts in securing CPS and networked control systems against spoofing (e.g., [2], [9], [11], [15], [16], [18]) and jamming/delaying attacks (e.g., [7], [13], [21]). These studies differ in their assumptions about the attacker's knowledge and the types of signals that can be attacked. In [9], the authors show false data injection attacks on state estimators in power grids. The idea is to craft an attack vector – in which each element corresponds to an injected measurement from a meter – that when combined with the state estimators, would still pass detection. The authors assume the attacker knows the configuration of the power system but may not have access to all the meters. The work in [11] generalizes false data injection attacks on control systems with specific controllers (e.g., Kalman) and shows that these attacks can cause the system to become unstable. In [16], the authors illustrate a scheme in which the control signals can be spoofed in tandem with the measurement signals to hide the effect of the attack. This scheme in effect hijacks the operation of the CPS. In terms of jamming attacks, the authors in [1] study the performance of a linear control system subject to various Denial of Service (DoS) attack models on the measurements and control signals (e.g., random, Bernoulli, constrained and general). Wireless jamming has been shown to cause severe effects that may cripple the entire system (e.g., [12], [19]).

To defend against these attacks, in [15], the authors construct a safety envelope from the measurements obtained under the normal operation of the system (without attacks). Attack detectors are then constructed that compare the measurements received during the operation of the system to the ones maintained by the safety envelope. The authors in [18] assume knowledge of the state of the system and derive correlation graphs without attacks to study how that information impacts decisions. In [2] the authors prevent an adversary from finding attack vectors through identifying two sets: a set of sensors to protect and a set of state variables that can be independently verified. While the authors in [17] investigate the combination and configuration of various defense mechanisms using stateless and stateful detection schemes for CPS, they do not consider a game-theoretic approach. In our work, we present a framework in which these defenses – in addition to other blocks – can be orchestrated through rigorous game-theoretic approaches. The use of game theory has been instrumental in advancing the state-of-the-art in security games and their wide range of applications (e.g., [23], [3], [14]) as it operates under the worst-case scenario. In [23], the authors propose a hybrid game-theoretic approach for resilient system control that also considers the system security and robustness. The scenario is stated as a cross-layer design in which two games are intertwined – a zero-sum differential game is used for robust control and a stochastic one is used for the design of the defense mechanisms. Unlike this work, our framework captures details about the exact blocks used, their effectiveness and impact on the signals propagated in the network. We utilize the Stackelberg formulation that resembles the models that screen for threats [3], [14]. Our model is different, however, as it focuses on stateless check blocks applied to the CPS with delay constraints. Furthermore, we consider stateful check blocks and develop a maximin value iteration approach that derives optimal randomized strategies.

III. The BLOC Framework

In this section, we present a general model for CPS, the adversary model, the check blocks, and the BLOC framework.

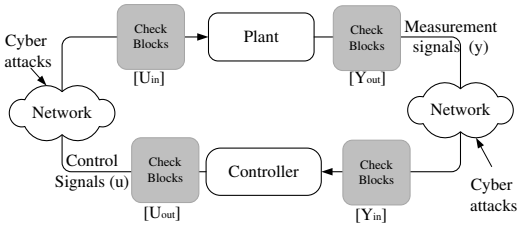


Fig. 1: A general block diagram for a CPS.

A. The CPS model

Figure 1 shows a generic block diagram of a CPS composed of a plant and a controller. Let x_k denote the state of the system at time k. A vector of measurement signals y_k is generated from the CPS and is fed to decision making entities (e.g., controllers that are centralized or distributed systems). The y_k signals intend to *capture* the state x_k in as representative way as possible, for instance by sensing the x_k values directly or by inferring the desired components from other dependent ones. The decision making entities process the y_k (measurement signals) and take control decisions u_k that change the state of the CPS. The dynamics of the plant and the controller can be captured through various models such as linear time invariant, linear time variant, non-linear or hybrid models. We assume that measurement signals and/or control signals traverse network components that are subject to cyber attacks.

B. The Adversary Model

We consider an adversary who is choosing different attack vectors on the measurement and control signals. If we let $\Gamma_u(u)$ and $\Gamma_y(y)$ denote the attack function on the control and measurement signals, respectively, then $\bar{u} = \Gamma_u(u)$ and

$\bar{y} = \Gamma_y(y)$ are the bad control and measurement signals received and acted upon by the CPS components. In case of a jamming attack, the attacked signal is simply omitted from the vector. Such "attacks" could just as easily be non-malicious but just as dangerous errors or unknowns, whether in the communication medium, in sensing, in actuation, or in some combination. We adopt a rational adversary model who is effectively selecting the $\Gamma_u()$ and/or $\Gamma_y()$ to bypass the check mechanisms in place to achieve the maximum damage.

C. The Check Blocks

Check blocks are the components that check signals and take actions (e.g., watermark, encrypt, alert, etc). In general, they can be divided into *stateless* blocks that operate instantaneously on the signal received, and *stateful* ones that maintain and utilize a state of previous values. Our models treat these checks as parameterized components in which their operations and parameters impact their effectiveness. For example, a cusum (cumulative sum for change detection) check block is an example of a stateful one that is required to keep a history and can have different effectiveness based on the threshold chosen; whereas a simple threshold check would be considered stateless. Check blocks include differentiators, aggregators, model predictors, state estimators, etc. as well as more sophisticated elements that utilize machine learning methods. We present examples of them within our BLOC framework.

D. The BLOC Framework

BLOC orchestrates the CPS control loop with various "check blocks" within four major locations as depicted in Fig. 1. The four major locations are: $[Y_{in}]$, $[Y_{out}]$, $[U_{in}]$ and $[U_{out}]$. Due to their unique locations, each one processes different types of signals—some are guaranteed to be valid (e.g., because of a tight coupling to a physical component), while some may be spoofed or distorted by noise.

- **The $[Y_{in}]$ location:** This location has access to all the previous measurement signals received $\bar{y}_1$, $\bar{y}_2$, ... $\bar{y}_k$ (which could have been spoofed or otherwise exhibit errors) and uses these historical values and the most recent measurements to calculate a current best estimate of the actual physical state. A common check that fits in this block is to measure the standardized residual using the χ^2 statistic which ensures that the received measurements do not deviate from the estimated state based on a threshold typically chosen based on a hypothesis testing criterion [20]. As illustrated in [9], such a check would not be enough to protect against spoofing attacks. Another component that fits in this location is a safety envelope test in which the new measurement signals received, $\bar{y}_k$, are compared against a measurement data model obtained under no attack using machine learning algorithms [15].
- **The $[U_{in}]$ location:** This location receives all the new control signals, $\bar{u}_1$, $\bar{u}_2$, ..., $\bar{u}_k$ (which could have been spoofed) and also has access to the current state of the physical system. It is also reasonable to assume that this component knows the initial state of the system, x_0, and thus can track the evolution of the state based on the control signals received (subject to the computational capabilities available). One of the primary goals of this block is to assert that the control signals received will not cause immediate violations of any of the system's target invariant properties if the actions embedded in the $\bar{u}_k$ are applied to the system (i.e., it simulates the result of applying the control signal). Because it has access to the historical state, this component can also verify properties relating to control signals changing too fast or too slow.
- **The $[Y_{out}]$ Location:** This location has access to all the previously generated measurements, y_1, y_2, ..., y_k and it can compare the new measurements to previous ones to detect immediate inconsistencies—in essence ensuring that the applied control did not violate any of the system's constraints. This location can also infer across elements within y_k over time to determine whether the vector of measurement signals is internally consistent. Finally, and perhaps the most useful against spoofing attacks, a check block can watermark y_k deliberately, so that the Y_{in} block can detect any tampering [5].
- **The $[U_{out}]$ Location:** This location can run/utilize a predictor (e.g., Model Predictive Control [4], [10]), given the current state estimate and the new control signal generated. Doing so, this block predicts the *expected state* of the physical elements of the system and verifies that this expected state does not violate any of the system properties. It assumes that a component in this location has access to the state space parameters, which is reasonable. Much of what is possible in this block is analogous to that in the $[U_{in}]$ block, but blocks in this location operate *before* the signals have traversed the network, where errors and attacks can happen.
  As in the $[Y_{out}]$ location, a block in the $[U_{out}]$ location can watermark the control signals and have a block in the $[U_{in}]$ location verify their presence to protect against spoofing attacks.

Within each of the above locations, "check blocks" are assigned based on the types of possible attacks, their effectiveness in detecting attacks, the overhead in running them and the importance of the signal(s) checked. Our game-theoretic assignment approaches are presented next for stateless (Sec. IV-A) and stateful (Sec. IV-B) blocks.

IV. Randomized Check Blocks Assignments

A. Stackelberg One-Shot games

We consider a game-theoretic formulation of a 2-player game between the defender and the adversary. The defender seeks to protect the CPS against various attacks through enabling a set of check blocks. The adversary seeks to attack the CPS through selecting a target signal (e.g., a measurement or a control signal) to attack and a particular attack method to employ (e.g., spoof, jam, delay). We assume a cost is incurred with each enabled block since it would operate on incoming

signals to detect attacks. Given a certain performance budget that the CPS can tolerate in the presence of these blocks, the defender seeks to assign blocks within the $[U]$ and $[Y]$ locations to maximize their utility subject to the adversary choosing her best response. In this problem we have:

- **Signal Targets** T: The adversary can choose to attack different target signals – measurements or control. The defender gains a utility U_t^p when signal t is protected and a utility U_t^u when signal t is unprotected.
- **Attacks** A: The adversary can choose an attack method a from a set of available attacks A to attack target t. We let c denote the attack category which is the tuple $\langle t, a \rangle$.
- **Check Blocks** B: The defender can choose to protect signals from attacks by enabling check blocks. We let E_b^a denote the effectiveness (i.e., the probability) of block b in protecting against attack a. Furthermore, we let T_b denote the time taken in executing block b to check a signal. We assume that we know the number of signals N_t that arrive under each target t.
- **Adversary Type** Θ: The model assumes the existence of different types of adversaries that may value targets differently.

We model this game as a Stackelberg one in which the defender commits first to an assignment that seeks to minimize the attacker's best response. We let $n_{b,t}$ denote the assignment of check block b to target signal t. The solution to the following optimization problem leads to such an assignment.

$$\max_{\mathbf{n}} \quad \sum_{\theta \in \Theta} z_\theta s_\theta \tag{1}$$

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

$$x_c = 1 - \Pi_{b \in B}(1 - E_b^a \times n_{b,t}) \qquad \forall c \tag{3}$$

$$\sum_{b \in B} n_{b,t} \times N_t \times T_b \leq C_t \qquad \forall t \tag{4}$$

$$n_{b,t} \in \{0, 1\} \qquad \forall b, t \tag{5}$$

The objective function 1 maximizes the defender's worst case possible utility, s_θ, given the probability z_θ of encountering an adversary of type θ. In equation 1, $\mathbf{n}$ represents the set of all feasible assignments of blocks to target signals. Constraint 2 enforces that the defender's utility, s_θ, is the worst possible over all possible attack categories $c \in C$ that the adversary could choose from. Equation 3 calculates the probability x_c of thwarting an attack of category c, given the assignment of blocks to targets $n_{b,t}$. Given that a subset of blocks are used to detect an attack a, the attack will go undetected only if all the blocks fail (hence, the product). Constraint 4 enforces that the performance of the set of enabled blocks to protect target t would not exceed a delay capacity C_t for any target t as dictated by the control loop. Equation 5 ensures that we have a valid assignment of each block being enabled or disabled.

To solve the above optimization problem, we perform two relaxations. Equation 3 presents a non-linear constraint which we simplify by considering the *most* effective block (rather than a function of all of them). Equation 5 makes the optimization model a Mixed Integer Program (MIP) and by replacing constraint 5 with $0 \leq n_{b,t} \leq 1 \quad \forall b, t$, we obtain a marginal assignment of blocks to targets. This marginal assignment is implementable in two ways: (1) The check blocks are not operating at their full scales (e.g., encrypting a signal with fewer bits, averaging over a smaller number of signals, or watermarking with fewer resolution), and (2) the block is using a sampling approach to only check a subset of the signals and that percentage is given by $n_{b,t}$ directly.

B. Markov Games

In our exposition thus far, we have assumed that once stateless blocks are applied, they become effective immediately. In many cases, however, blocks may need to observe the target signal(s) *over time* to detect attacks (i.e., to warm-up). For example, consider a spoofing attack on a HVAC system in which the measured temperature is off by a small value. This attack would not be instantaneously detected since it can pass as legitimate noise, but may cause the system to continuously operate. Thus, we let W_b denote the maximum warm-up period (i.e., number of steps needed) for block b to be effective which is given by:

$$W_b = \frac{R_b}{\min_{\mathbf{t} \in \mathbf{T}} F_t \times P}, \tag{6}$$

where R_b is the number of required signals by block b, F_t is the signal transmission rate for target t and P is the duration of a time-step in seconds. To capture environmental uncertainty in terms of the signals transmission rates due to the transient and steady-state operations of the system, we let the warm-up period be a random variable with a maximum value of W_b. As a proxy for C_t and to capture the delay constraints dictated by the control loop, we assume each target has a capacity in terms of the maximum number of assigned blocks.

To address the statefulness property, we model the problem as a two-player zero-sum Markov game, in which a sub-game is played in every time-step. The game is represented by the tuple $\langle \mathcal{S}, \mathcal{A}_a, \mathcal{A}_d, \mathcal{T}, \mathcal{R}, \beta \rangle$ where:

- $\mathcal{S}$ is the finite set of system states, where each $s \in \mathcal{S}$ describes the assignment of blocks to targets as a matrix in which rows corresponds to targets and columns to blocks. The value of each element indicates its remaining warm-up time until the block becomes effective or if the block is not deployed. A value of zero indicates that the block has become effective. Since the warm-up time is bounded for all blocks, $|\mathcal{S}|$ is bounded by the number of possible combinations of these values raised to the power of the number of targets.
- The attacker has a finite set of actions ($\mathcal{A}_a$) – each corresponds to a different attack method to be played in every sub-game. Conversely, $\mathcal{A}_d$ is the finite set of actions of the defender. We let $\mathcal{A}_d(s)$ denote the available actions in state s. Actions include adding or removing one block, or making no changes to the current block assignment.
- $\mathcal{T} : \mathcal{S} \times \mathcal{A}_a \times \mathcal{A}_d \times \mathcal{S} \Rightarrow \Pi(\mathcal{S})$ is the state evolution function based on the agents action pair, where Π is a

discrete probability distribution over $\mathcal{S}$. Accordingly, we let $\mathcal{T}(s, a, d, s')$ denote the probability of transitioning into state s' from state s when the attacker and the defender take actions $a \in \mathcal{A}_a$ and $d \in \mathcal{A}_d$, respectively.

- $\mathcal{R} : \mathcal{S} \times \mathcal{A}_a \times \mathcal{A}_d \times \mathcal{S} \Rightarrow \mathbb{R}$ is the reward obtained by the defender. We let $\mathcal{R}(s, a, d, s')$ denote the reward received going from s to s' under actions $a \in \mathcal{A}_a$ and $d \in \mathcal{A}_d$.

$$\mathcal{R}(s, a, d, s') = \sum_{t \in T} x_{a,t,d} U_t^p + (1 - x_{a,t,d}) U_t^u, \quad (7)$$

where $x_{a,t,d}$ is the effectiveness of assignment d against attack action a for target t (which is computed as in Equation 3). Due to the zero-sum nature of the game, one reward function suffices for both agents.

- β : is the discount factor such that $0 < \beta < 1$.

To derive optimal strategies we apply value iteration [8] on the Markov game. Equations 8-9 describe the value iteration over the states, in which $V(s)$ is the expected reward for the defender when starting from state s and following the optimal policy thereafter. The quality function $Q(s, a, d)$ is the immediate reward obtained from action pair a and d, plus the expected discounted future reward the defender expects to receive over the successor states following the optimal policy.

$$Q(s, a, d) = \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, d, s') R(s, a, d, s') + \beta \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, d, s') V(s'), \quad (8)$$

$$V(s) = \max_{\pi \in \Pi(\mathcal{A}_d)} \min_{a \in \mathcal{A}_a} \sum_{d \in \mathcal{A}_d} \pi(s, d)\ Q(s, a, d). \quad (9)$$

In every sub-game, we generate a payoff matrix with dimensions defined by the players' action spaces. Each entry holds the quality value of each action pair. We then solve that sub-game by finding Nash equilibrium whose pure or mixed strategy, $\pi(s, d)$, is the probability distribution over the available actions. The linear program described in Equations 10 - 13 guarantees the worst-case payoff z over the attacker's available actions, while ensuring a valid distribution over the defender's actions.

$$\max_{\pi(s,d) \in \Pi(\mathcal{A}_d)} z \quad (10)$$

$$z \leq \sum_{a \in \mathcal{A}_a} \pi(s, d) Q(s, a, d) \quad (11)$$

$$\sum_{d \in \mathcal{A}_d(s)} \pi(s, d) = 1 \quad (12)$$

$$0 \leq \pi(s, d) \leq 1 \qquad \forall d \in \mathcal{A}_d(s) \quad (13)$$

V. Performance Evaluation

In this section, we present the evaluation of our game-theoretic block assignments.

A. Stateless Check Blocks Assignments

To assess the performance of our Game-Theoretic approach (captured by the optimization problem presented in Equations 1-5 and the relaxations described in sub-section IV A), we compare its outcome to four other approaches: Random, Greedy, Greedy-LP, and the game-theoretic Mixed Integer (GT-MI) approach. In the Random approach, a random assignment of check blocks is performed while still satisfying the capacity constraints in Equation 4. In the Greedy approach, a weighted effectiveness for each block is calculated by dividing its effectiveness by its execution time. The total effectiveness of the blocks is then computed by summing their weighted effectiveness over the different attacks they cover. The blocks are then sorted in a list by their total weighted effectiveness in descending order. We assign the first block to the most valuable target at the the highest possible coverage (subject to the capacity constraints) and consecutively to the remaining less valuable targets. The procedure continues as long as the capacity constraints are not violated. In the Greedy-LP approach, a mathematical program (not presented due to space limitation) is solved with the objective to maximize the defender's utility when targets are undefended. We used the same capacity, *most* effective block, and partial assignment constraints, but added a new constraint that ensures that each target has a minimum coverage. We labeled the approach as Greedy-LP because in this model the integrality on assignment constraints is relaxed. Finally, in the GT-MI approach, we let the check blocks be either fully assigned or not and solved the model using the Branch and Cut algorithm.

We consider a CPS composed of 6 target signals that are subject to 3 attacks from 1 type of adversary. The defender has 3 check blocks to enable. We consider 3 levels of effectiveness for the check blocks: low, medium and high, that are chosen uniformly at random below 0.5, between 0.5 and 0.7 and above 0.7, respectively. To study the impact of the importance of the target signals, we consider 3 cases of utilities: low, medium and high. We kept the total utility fixed at 700 to compare between the cases. In the low utility case, all the targets have an equal value of 116.67. In the medium case, one of the targets has a higher utility randomly chosen between 200 and 300 and the rest have equal values. In the high case, one target has a higher utility randomly chosen between 300 and 450 and the rest have an equal value. Thus, we had a total of 9 experimental cases. In each case, the results are computed and averaged over 5 independent runs.

Fig. 2 presents the average worst case utility for different effectiveness and utility levels. Fig. 2 (left), (center), and (right), presents the average worst case utility, under low, medium and high utility cases, respectively. One can see that across all the cases our game-theoretic assignment achieves the best worst-case followed by the mixed integer approach. The random approach gives the lowest worst-case utility in all cases. In the low and medium utility level cases, Greedy-LP achieves better results than the Greedy approach, whereas in the high level of utility, Greedy achieves better results than Greedy-LP. Since Greedy-LP aims to increase coverage across targets, it performs well when the variance in the utilities is small (i.e., in the low and medium utility cases). Greedy, on the other hand, seeks to protect the most valuable targets, and thus may leave one (or more) targets undefended which results

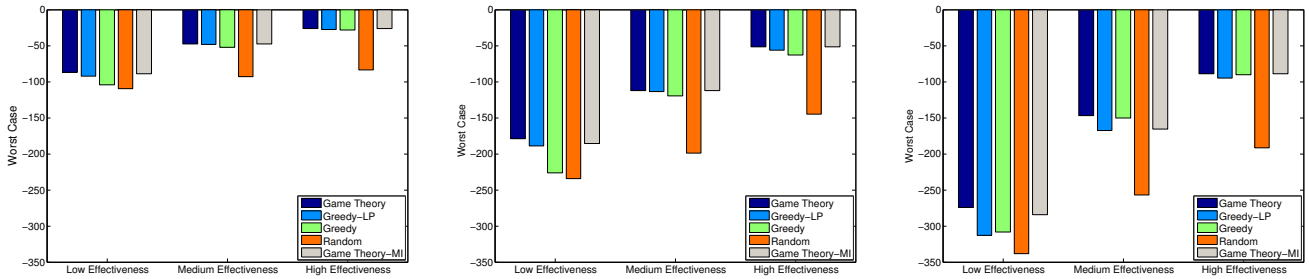


Fig. 2: Impact of different assignment strategies under low (left), medium (center) and high (right) utilities.

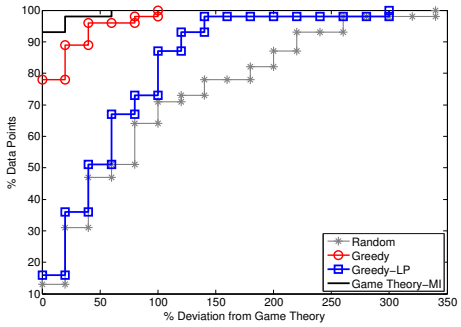


Fig. 3: Performance profile between defender strategies.

in a better worst-case over Greedy-LP at high levels of utility.

Fig. 3 shows the relative performance of the Game Theory-MI, Greedy, Greedy-LP and Random in comparison to our optimal Game-Theoretic approach. The x-axis represents $\%$ of absolute error in worst-case utility and y-axis represents $\%$ of data points that fall within that $\%$ of error. For example, if we compare the Greedy approach with Game Theory, we can observe that approximately 77% of data points (Worst-case utility) are within 20% deviation from the Game Theory approach. Random and Greedy-LP gave the same worst-case utility as Game Theory in less than 20% of the experiments. Also, in nearly 95% of the experiments, Greedy has below 50% error. For Greedy-LP and Random, the percentage of instances with error below 50% is only about 40-50%.

B. Stateful Check Blocks Assignments

To assess the behavior of our game-theoretic approach in comparison to other strategies in the presence of stateful blocks, we conducted extensive experiments on a number of scenarios for a system with 3 targets, 4 blocks and 3 types of attacks. Each scenario is described by a maximum warm-up vector $[W_1, W_2, W_3, W_4]$, which we chose from the set $\{[1,1,1,8],[1,2,3,4],[2,2,2,2],[1,2,4,8]\}$ and a block capacity vector $[C_1, C_2, C_3]$, which we chose from the set $\{[3,2,1],[2,2,2],[3,3,3]\}$. For example, the scenario $[1,2,4,8]-[3,2,1]$ means that the 4 blocks require 1, 2, 4 and 8 maximum time-steps to be effective, with the 3 target signals having capacities corresponding to 3, 2 and 1 maximum assigned blocks. In each scenario, we studied 9 settings with low, medium, and high effectiveness in conjunction with low, medium, and high utilities (computed similar to the previous section). Additionally, we generated the warm-up values randomly based on a geometric probability distribution in the range $[0, W_b)$, where 0 (no warm-up time) is the least probable outcome.

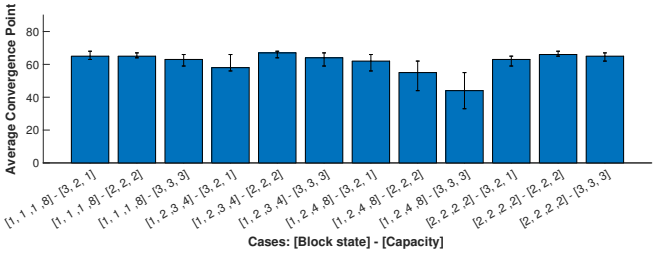


Fig. 4: Average number of iterations until convergence.

We first trained both agents against each other through the value iteration algorithm. We used a discount factor β of 0.99 and the convergence point is assumed to be the iteration, in which the change in the average value function is less than 1 percent of the previous value. Fig. 4 shows the average number of iterations until convergence over the 9 settings.

Once the agents were trained, we simulated the system and assessed the behavior of the rational strategy against random, greedy and myopic strategies. The rational strategy picks actions based on the learned policy. The myopic strategy picks actions based on the immediate rewards in the current sub-game. The random strategy picks actions at random. The greedy defender strategy picks actions that assign the most effective block to the highest utility target, while the greedy attacker strategy picks the attack action that is the least detectable.

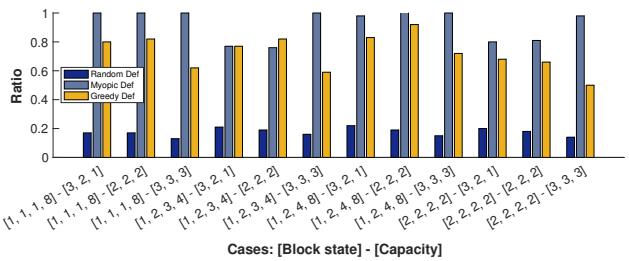


Fig. 5: Relative behavior of rational defender to other methods.

To assess the relative performance of the defense strategies in comparison to the rational one, Fig. 5 shows the final payoff ratio averaged over 50 runs under the high effectiveness - high utility setting. One can see that the random policy is drastically worse than the rational policy. The rational policy outperforms the greedy approach, because the greedy defender does not take into account the presence of the attacker. Instead, it assigns blocks based on their average effectivenesses. Also, even though the greedy defender may find a good block assignment, it leaves the system vulnerable during the intermediate time-steps toward reaching that assignment. This becomes more notable as the capacities of the targets increase. Myopic policy also loses to rational strategy, as it may remove blocks that are not effective yet, looking for a higher immediate reward. This translates into playing a random policy if none of the blocks are going to be effective in the next time-step.

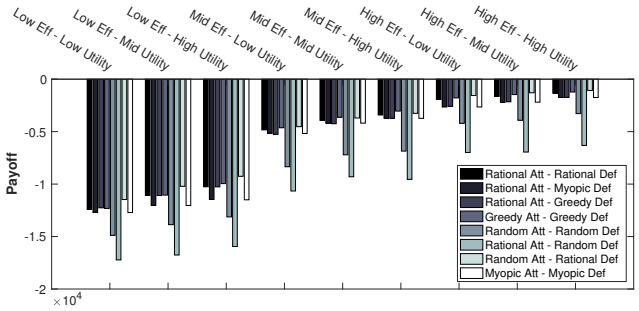


Fig. 6: Payoff of various strategies under different settings.

To investigate the impact of the 9 experimental settings (effectivenesses and utilities), we consider the scenario $[1, 2, 3, 4] - [3, 2, 1]$. Fig. 6 shows the various strategies' payoff for both players. Under low effectiveness, deviating from the rational policy does not have a significant impact on the final payoffs, because the variance in the actions' reward values is low. However, under high effectiveness, the differences between policies are significant, which makes playing rationally crucial. Having highly effective blocks in real settings requires playing rationally to optimally protect the system. Furthermore, experiments with different utilities produce a similar effect, but is not as notable as with the effectiveness. Fig. 7 represents the behavior of different defense policies under the $[1, 2, 3, 4] - [3, 2, 1]$ scenario, where each point is the average payoff each agent attains in every time step over 50 runs. It is clear that the rational defender has the best final payoff in comparison to other policies. Also, the rational attacker outperforms other attack strategies in all cases.

C. Implementation results from a real HVAC test-bed

1) The Setup: We orchestrated a standard HVAC system with an implementation of BLOC as depicted in Fig. 8. The system is set up with a Raspberry Pi, which we will refer to as the controller, connected to the HVAC system in place of a typical thermostat. A DS18B20 temperature probe is wired to the controller that supplies the current temperature up to an accuracy of 5 digits. The controller then uses the temperature to generate control signals to send to the HVAC.

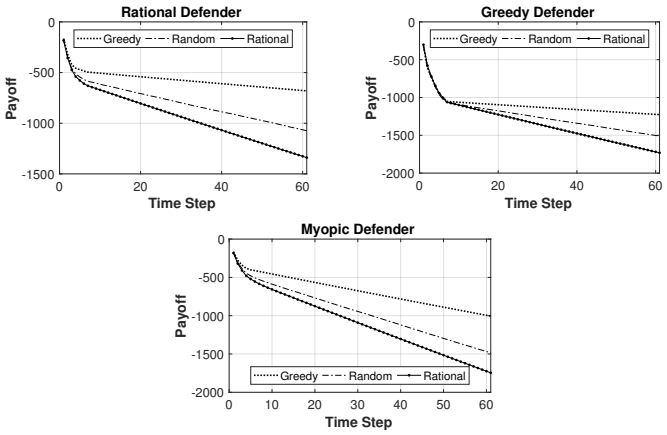


Fig. 7: Average payoff over time for various defense strategies.

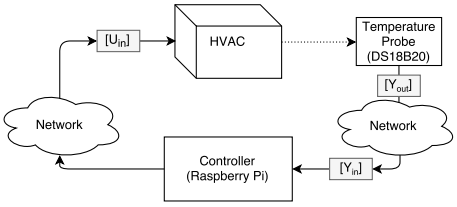
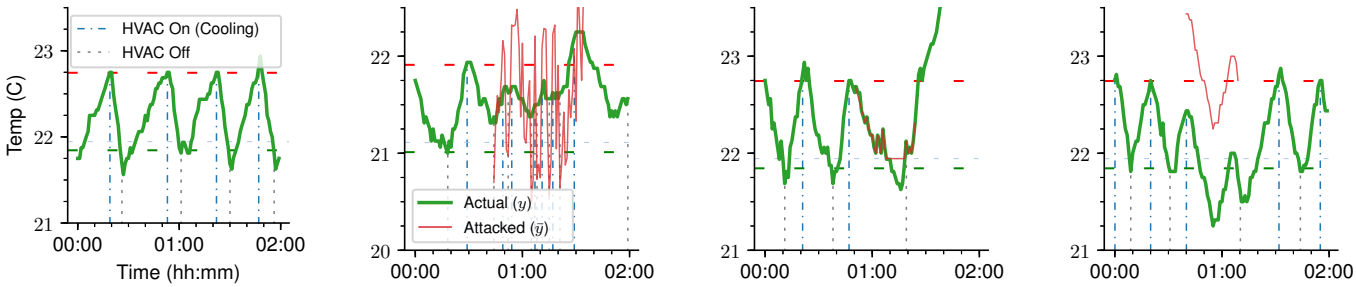


Fig. 8: HVAC Setup

Fig. 9 (a). shows the operation of the system under normal circumstances when the AC was used for cooling. The target temperature is set to $21.94°$ Celsius ($71.5°$ F), and the controller is configured to turn on when the temperature reaches a delta of $0.8°$ C from the target and turn off once it reaches a delta of $0.1°$ C (overshooting the target).

2) Stealthy Attacks: We consider a stealthy adversary that introduces artifacts in the HVAC system's stability by increasing the number of events (e.g., switching the AC on and off more often than necessary) and/or extending the AC running period. We assume the attacker has already defeated any security measures in-place and is able to directly modify the actual temperature y from the probe and report $\bar{y}$ instead. We consider 3 types of attacks: fuzzy, clamp, and offset.

- **Fuzzy attack**: $\bar{y} = y + \text{unif}(k_a, k_b)$
  This attack adds a random offset in the range of $[k_a, k_b]$ to the input temperature to coerce the controller into sending incorrect control signals to the HVAC system. Fig. 9 (b) shows the effect of this attack with the range $[-1, 1]$. The number of events increased by almost 4 times over the period of the attack with the AC running around twice as long as it would have under normal circumstances.
- **Clamping attack**: $\bar{y} = \begin{cases} y & y > k \\ k & y \leq k \end{cases}$
  This attack clamps the temperature to an arbitrary minimum k, to make the HVAC work more than necessary. Fig. 9 (c) shows the effect of this attack when $k = 21.94°$. The attack is enabled at around 0:40 minutes in, and disabled at around 1:25. One thing to note is that the effects of this attack are felt even after it is disabled,



(a) (b) (c) (d)

Fig. 9: Normal Operation (a); Fuzzy Attack: range = $[-1, 1]$ (b); Clamp Attack: $k = 21.94$ (c); Offset Attack: $k = 1$ (d).

as it causes the HVAC system to overstrain itself, which eventually makes it completely ineffective for a period of time lasting around 30 minutes.

- **Offset attack**: $\bar{y} = y + k$
  This attack adds a chosen *constant* offset k in the range of $[0, 1]$ to the input temperature, effectively changing the target temperature by $-k$. Fig. 9 (d) shows the effect of this attack when $k = 1°$. This particular attack caused the AC to continuously work while the attack was active.

	Normal	Fuzzy	Clamp	Offset
Control Events	100%	375%	50%	0%
Time Running	26.9%	55%	100%	100%

TABLE I: Impact of each attack (as shown in the graphs).

Table I shows the impact on the percentage of control events and AC run time during attacks.

3) The BLOC Defense: We used 3 types of check blocks – two stateful and one stateless – to protect against the attacks. For the stateful blocks, the state is defined by its warm-up time. When an attack is detected, an alarm is raised to the operator. Each block is assigned a position on the control loop and its effectiveness is driven by its likelihood to detect particular attacks as we explain in detail below and show in Table II.

	Fuzzy	Clamp	Offset
Control Frequency (state=6)	0.99	0	0
Delta Temp. 1 (state=2)	0.7	0	0.8
Delta Temp. 2 (state=5)	0	1	0
Watermark (stateless)	0.9	0.9	0.9

TABLE II: Effectiveness of blocks against attacks.

- **Control Frequency Block** - $[U_{in}]$: This block measures the frequency at which control signals (on/off signals) are sent to the HVAC. An alarm is raised if the rate of control events exceeds a certain threshold (6 per hr). Under fuzzy attacks, we calculated the effectiveness in the worst-case scenario, wherein the current temperature is situated evenly between the two activation thresholds. In this spot, there is a significant chance every time-step that the fuzzy attack will trigger a new control event. We then calculated the probability that the fuzzy attack will trigger 6 or more control events in an hour. Both the clamp and offset attacks do not increase the rate of control events, so we assumed an effectiveness of zero.
- **Delta Temperature Block** - $[Y_{in}]$: This block measures the delta between the previously recorded temperature and the current temperature. Two delta temperature blocks have been added to the system. The first block defines the maximum delta as $0.3°$c per minute. This threshold was chosen as it resulted in 0% false positive detection rate under normal operation. The second block defines the minimum delta as $0.05°$c per every 5 minutes. We computed the effectiveness against fuzzy attacks by taking the percentage of values that can be chosen that are greater than ± 0.3.
- **Watermark Block** - $[Y_{out}] \rightarrow [Y_{in}]$: This block adds a checksum digit to the end of the 5-digit temperature reported by the probe to create a 6-digit temperature. The change in temperature from inclusion of this block is insignificant and can be ignored. This block has the effect of being able to detect 90% of any type of attack, because each attack has a 1 in 10 chance of getting the checksum digit correct.

4) The Markov Game (BLOC vs. Adversary): For our test of the system, we set up a rational BLOC defender against a rational attacker playing a Markov game with a sub-game in every time-step (where a time-step is one minute).

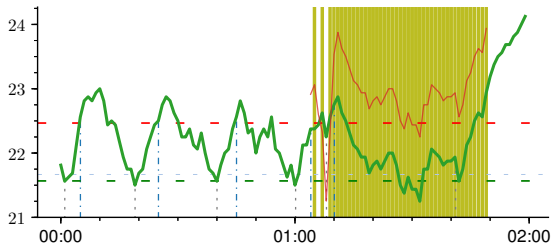


Fig. 10: Rational defender vs. rational attacker.

Fig. 10. shows the results of our test. The attack is launched a few minutes after the first hour, and left running for approximately 45 minutes. When an attack is detected, a thick vertical line is plotted on the graph. The attacker first cycles through his attacks and then after 8 minutes settles on the offset attack (worst-case). This particular offset chosen happens to

be detectable by the watermark block, as shown by the solid-shaded portion of the graph. Note that a greedy, random, or other form of defenses would be defeated by the attacker since it can choose the worst case attack to be mounted based on its knowledge of the check blocks used. Our game-theoretic approach, on the other hand, drives the optimal strategy to minimize the worst-case damage.

VI. Discussion and Conclusion

Cyber attacks on CPS are becoming more sophisticated and the defense mechanisms to check against have not just become largely specific – but increasingly more computationally expensive (e.g., using deep learning). This work presents a coherent framework – BLOC – that orchestrates the CPS at runtime with the proper "check blocks" that are assigned through game-theoretic approaches. From a security standpoint, our approaches are commendable as they operate under the worst-case scenario (which is exactly what our maximin strategy optimizes for). Additionally, they are randomized through the mixed-strategy of the defender and operate within the delay bound dictated by the the CPS control loop.

We have demonstrated the superiority of our game-theoretic approaches over a wide range of policies using stateless and stateful blocks. In the stateless case, we formulated a Stackelberg game and used linear and mixed integer programming to obtain optimal policies. Our evaluation shows that conventional approaches such as greedy (and its LP variant) and random are always worse than the game-theoretical policy. In the stateful case, we developed a Markov game model and derived optimal policies using value iteration approach. Despite the combinatorial nature of the problem (e.g., the underlying Markov game can reach 8 million nodes and 500 million edges), we were able to obtain optimal mixed strategies. Our results show that the derived policies outperform myopic, greedy and random polices. They perform better than myopic as they can account for the effectiveness of the blocks in the future (due to their warm-up periods) whereas the myopic policy is biased toward higher immediate rewards. They perform better than greedy as the greedy policy does not account for the adversary and assigns blocks with the highest effectiveness. We observed that even in the cases when both policies agree to the same assignment, our game-theoretic policy intelligently protects the system in *intermediate steps*. In some cases it would assign blocks with shorter warm-up periods and lower effectivenesses to keep the system secure until more effective blocks have warmed up that later would replace less effective ones. Based on our extensive simulations with various parameters for effectiveness and target utilities, we have found that the worst-case payoff of the defender under random, greedy, and greedy-LP were between $[20\%, 69\%]$, $[0\%, 29\%]$, and $[0\%, 20\%]$, respectively, worse than the payoff achieved by the game-theoretic policy with stateless blocks. Similarly, with stateful blocks, the rational defender outperforms the random, greedy and myopic defenders by attaining better worst-case payoffs in the ranges of $[26\%, 87\%]$, $[0\%, 50\%]$ and $[0\%, 24\%]$, respectively.

Acknowledgments

This research is funded in part by NSF CNS awards #1149397 and #1239498.

References

[1] S. Amin, A. Cárdenas, and S. Sastry. Safe and Secure Networked Control Systems under Denial-of-Service Attacks. *Hybrid Systems: Computation and Control*, pages 31–45, 2009.

[2] 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.

[3] 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.

[4] E. Camacho and C. Alba. *Model Predictive Control*. Springer Science & Business Media, 2013.

[5] R. Chabukswar. Secure Detection in Cyberphysical Control Systems. *Ph.D. Thesis – CMU*, 2014.

[6] J. Finkle. U.S. official sees more cyber attacks on industrial control systems, 2016.

[7] J. Hespanha, P. Naghshtabrizi, and Y. Xu. A Survey of Recent Results in Networked Control Systems. *Proceedings of the IEEE*, 95(1):138–162, 2007.

[8] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the eleventh international conference on machine learning*, volume 157, pages 157–163, 1994.

[9] 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.

[10] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6), 2000.

[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] K. Pelechrinis, M. Iliofotou, and V. Krishnamurthy. Denial of Service Attacks in Wireless Networks: The Case of Jammers. *IEEE Communications Surveys & Tutorials*, 13(2), 2011.

[13] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. Sastry. Foundations of Control and Estimation Over Lossy Networks. *IEEE*, 95(1):163, 2007.

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

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] M. Wilhelm, I. Martinovic, J. Schmitt, and V. Lenders. Short Paper: Reactive Jamming in Wireless Networks: How Realistic is the Threat? In *ACM conference on Wireless network security*, 2011.

[20] A. Wood and B. Wollenberg. *Power Generation, Operation, and Control*. John Wiley & Sons, 2012.

[21] T. Yang. Networked Control System: A Brief Survey. *IEE Proceedings Control Theory and Applications*, 153(4):403–412, 2006.

[22] X. Zheng, C. Julien, M. Kim, and S. Khurshid. Perceptions on the state of the art in verification and validation in cyber-physical systems. *IEEE Systems Journal*, 11(4):2614–2627, 2017.

[23] Q. Zhu and T. Basar. Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: Games-in-games principle for optimal cross-layer resilient control systems. *IEEE control systems*, 35(1):46–65, 2015.