2021 IEEE Conference on Control Technology and Applications (CCTA).
August 8-11, 2021. San Diego, California

2021 IEEE Conference on Control Technology and Applications (CCTA).
August 8-11, 2021. San

A Human-Integrated Tool for Proactive and Reactive Security in Cyber-Physical Systems

Alistair J. Sequeira, Aris Kanellopoulos and Kyriakos G. Vamvoudakis

2021 IEEE Conference on Control Technology and Applications (CCTA) | 978-1-6654-3643-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/CCTA48906.2021.9659220

***Abstract*— We present a comprehensive defense software for cyber-physical systems, comprising both proactive and reactive mechanisms. Specifically, the demonstrated tool allows a human operator to remain aware of the system's health and operation, while an autonomous subsystem applies a switching rule based on the principles of Moving Target Defense, rendering the system more unpredictable but stable nonetheless. Finally, the man-machine interaction implements a trust metric, that allows either the autonomous mechanism or the human agent to have more control over the system based detection and mitigation history. We describe the interface of the security software, while the case of an autonomous aircraft is used to showcase the efficacy of the software.**

I. Introduction

Due to the increasing complexity and their widespread use in various domains, cyber-physical systems (CPS) have become a prime target for various attacks whose aim goes beyond the cyber domain [1]. This, in turn, has lead to an increased need for security tools that retain intuitive interfaces that enable a security analyst to have complete awareness of the system's operation. Autonomous defense systems typically include detection and mitigation algorithms operating in a reactive fashion, i.e., by responding to the injected signal by the attacker. Different from this approach, proactive defense mechanisms seek to make the system intrinsically harder to affect. One such approach, Moving Target Defense (MTD), rests on the idea that by dynamically shifting certain system parameters, it becomes prohibitive for the attacker to achieve reconnaissance of the system. This increased unpredictability is considered as a deterrent for the adversaries. While security for CPS has been an active research topic in recent years, there is still need for the development of tools that integrate human analysts with autonomous defense mechanisms guaranteeing optimal cooperation between human and machine.

Related Work

A survey of results on attack detection on smart grids is presented in [2], where the algorithms remain model-based. Our previous work in [3] introduced an Integral Bellman-based detection system for sensor and actuator attacks that may be used in tandem with reinforcement learning methods in a model-free fashion [4]. The theoretical foundations of proactive defense methods were presented in [5], while our previous work in [3] employed the same principles towards a switching-based MTD framework. The cooperation between man and machine based on trust metrics has been investigated extensively. In [6] the authors investigate the structural parameters that affect trust in a static model. In [7], the authors take a dynamic approach and detect trust changes based on the number of times that the human was forced to intervene. Our tool integrates the method proposed in [8], where a Bayesian model was used to track fluctuations in trust between human agents and autonomous systems.

Contribution: The contributions of the present work comprise of a tool that is able to simulate various attacks and test the robustness of an Autonomous System (AS), based on MTD, together with a human-in-the-loop, which can be an analyst in training or part of an already deployed defense system. In accordance with MTD, we design multiple controllers for every admissible combination or 'modes' of actuators. Furthermore, to ensure appropriate levels of collaboration between the human and AS, a trust metric is proposed. The trust metric data is then used to advise the human to either take or return control to the AS through the UI.

Notation: The notation used here is standard. $\bar{\lambda}(A)$ is the maximum eigenvalue of the matrix A and $\underline{\lambda}(A)$ is its minimum eigenvalue. $\|\cdot\|$ denotes the appropriate vector norm. The transpose of a matrix is denoted as $(\cdot)^{\mathrm{T}}$. The cardinality of a set is denoted by $\mathrm{card}(\cdot)$. Furthermore, $\mathrm{randi}(\cdot)$ represents a random integer chosen within a range and $\mathrm{rand}(\cdot)$ represents a randomized, normalized vector. Finally $\wedge, \vee$ and $\veebar$ represent the 'logical and', 'logical or' and 'exclusive or' respectively.

II. User Interface (UI) Description

The tool is an application user interface (UI), as seen in Figure 1, that provides information on trajectory tracking via the "System State Trajectory Data" Graphs 1 and 2, the time history of most recently used modes with the "System Active Mode" graph and the "Autonomous System (AS) Integral Bellman Error" Graph, based on [3]. Additionally a trust metric is introduced as seen in the "Trust Performance" Graph.

The user is also provided with real-time data as seen in the top left corner of the UI, and currently active actuators data in the bottom left corner, under "Active Actuator for System States". Each actuator will be lit green (supported by '+1') if active, red (supported by '-1') if timed out (based on potential attack) or transparent (supported by '0') if available for use.

A. J. Sequeira, A. Kanellopoulos and K. G. Vamvoudakis are with the Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA e-mail: (asequeira6@gatech.edu, ariskan@gatech.edu, kyriakos@gatech.edu).

This work was supported in part by ONR Minerva under grant No. N00014-18-1-2160, by NSF CAREER under grant No. CPS-1851588, and by ARO under grant No. W911NF-19 − 1 − 0270.

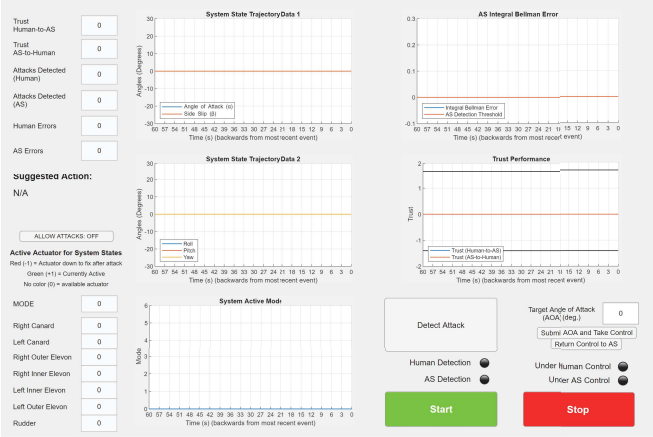

Fig. 1. User Interface of the tool.

The bottom right corner of the UI holds the "Detect Attack" Button for the user to click, and the two lights below it indicate whether AS and the Human made the correct detection based on a green/red color respectively.

The feature next to the "Detect Attack" Button is the opportunity for the user to take control of the system. Specifically, for this ADMIRE aircraft system [9], the control is only applicable for the angle of attack (AOA) of the aircraft. The respective buttons below the AOA input, allow the user to take control by directing the aircraft to a certain AOA. Correspondingly, the "Under Human Control" or "Under AS Control" lights will turn green when the respective entity is in control of the aircraft's AOA.

The last few features of the UI are the ability for the user to switch on/off the attacks on the system via the "ALLOW ATTACKS" button and a warning mechanism of "Suggested Action" to advise on when the user should take/give up control, whose mechanism will be discussed in further detail in the following section.

III. Tool Breakdown

The underlying components of the UI are based on the provided Algorithm 1 and consist of the following steps $\forall t$:

1) Generate attacks on system.
2) Get desired trajectory from user input of UI.
3) Apply control to the system via the control system that accounts for both human and AS input.
4) Update trust metrics of Human and AS based on propagated dynamics and UI input of detection of attacks.

Following [3], we consider the following linear continuous-time system,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B_i u(t),\ t \geqslant 0, \\ y(t) &= Cx(t), \end{aligned} \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is the state, $u(t) \in \mathbb{R}^m$ is the potentially attacked input of the system, $y(t) \in \mathbb{R}^p$ is the output, $A \in \mathbb{R}^{n\times n}$ is the plant matrix. The set of matrices $\{B_i\}$, $B_i \in \mathbb{R}^{n\times m}$, $\forall i$ comprises different actuator combinations which can be switched in real time, while $C \in \mathbb{R}^{p\times n}$ is the output matrix. Formally defined, B_i belongs to the set of candidate actuating modes, while we also define those modes B_c which determine the system fully controllable [3]. The currently active mode from B_c is labelled as $s(t)$.

A. Attack Generator (Step 1)

Each attack on the system, generated at regular predefined intervals, has a 'dwell' time, indicating the duration of the attack, and a 'rest' time, indicating the time before attacks on a specific mode. Considering this for all $\text{card}(B_c) := N_m$ modes, the dwell time and rest time are defined as,

$$\begin{aligned} t_{\text{Dwell}}(t) &= \{t_{D_i}(t) \mid i \in \{1, \ldots, N_m\}\}, \\ t_{\text{Rest}}(t) &= \{t_{R_i}(t) \mid i \in \{1, \ldots, N_m\}\}, \end{aligned}$$

where $t_{D_i}(t)$ and $t_{R_i}(t)$ are the attack and rest duration times for the i^{th} mode at time t and are defined as,

$$t_{D_i}(t) = t_{D_i}(t-T) - T(t_{D_i}(t-T) > 0) + D_{\text{set}_i}, \tag{2}$$

$$t_{R_i}(t) = t_{R_i}(t-T) - T(t_{R_i}(t-T) > 0) + R_{\text{set}_i}, \tag{3}$$

where D_{set_i}, derived by the equation below, is the opportunity to set off an attack on the i^{th} mode, based on a number of factors to be discussed. Similarly, R_{set_i} also operates in the same manner, but its conditions depend on whether the attack on the i^{th} mode is ending.

$$D_{\text{set}_i} = \begin{cases} \text{randi}([T_{D_L}, T_{D_H}]), \\ \quad \text{if randi}([1, N_a]) > 1 \wedge s(t-\delta t) = i \\ \quad \wedge(\sum_{i=1}^{N_m}(t_{D_i}(t-T) > 0) < N_{\max}) \\ \quad \wedge t_{D_{s(t)}}(t-T) \leqslant 0 \wedge t_{R_{s(t)}}(t-T) \leqslant 0, \\ 0, \quad \text{otherwise}, \end{cases}$$

$$R_{\text{set}_i} = \begin{cases} \text{randi}([T_{R_L}, T_{R_H}]), & \text{if } 0 < t_{D_i}(t-T) \leqslant T, \\ 0, & \text{otherwise}, \end{cases}$$

where $N_a > 1$ is some integer constant used to control the chance of attacks occurring to at least 50% or higher. And $N_{\max}$ is given by (4), noting that N_{lim} is user specified with the purpose of limiting the attacks on the system so as to not overwhelm the human.

$$N_{\max} = \begin{cases} N_{\text{lim}}, & N_{\text{lim}} < N_m, \\ N_m, & \text{otherwise}. \end{cases} \tag{4}$$

Finally, the attacked control input $u_a(t)$ and its type $u_t(t)$ are given by (5) and (6), where $u_t(t) = 1$ implies $u_a(t)$ will replace original control $u(t)$ and $u_t(t) = 2$ implies $u_a(t)$ will add to $u(t)$ and f_a is a scale factor for the size of the attack.

$$u_a(t) = \begin{cases} f_a\ \text{rand}([\mathbb{R}^m]), & t_{D_{s(t-\delta t)}}(t) > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

$$u_t(t) = \begin{cases} \text{randi}([1,\ 2]), & t_{D_{s(t-\delta t)}}(t) > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{6}$$

where $\text{rand}([\mathbb{R}^m])$ creates a normalized random vector of the same size of u(t). For the tool demonstration seen throughout this paper, values of $N_a = 3$, $N_{\text{lim}} = 2$ and $f_a = 1$ were used.

B. Desired Trajectory (Step 2)

The desired trajectory is only considered when the human is controlling the system. For the system developed by [9], requiring only the desired angle of attack AOA_{input} as a human input, would yield the desired trajectory given by $x_{\text{ref}}(t) = \begin{bmatrix} AOA_{\text{input}} & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$.

C. Control System (Step 3)

The Control System is broken into two main parts of human control and AS control. The AS control is taken from [3]. The AS control is further broken into two parts of the proactive mode switching and AS detection.

1) Proactive Switching and AS Control: Each switching mode corresponds to an optimal feedback controller, $u_i^\star(t) = -K_i x \ \forall x$, where,

$$K_i := R_i^{-1} B_i^{\mathrm{T}} P_i, \tag{7}$$

and each matrix P_i is found as a solution to a Riccati equation,

$$0 = A^{\mathrm{T}} P_i + P_i A - P_i B_i^{\mathrm{T}} R_i^{-1} B_i^{\mathrm{T}} P_i + Q_i. \tag{8}$$

The proactive switching is then conducted according to a probability given as,

$$p_i = e^{\left(-\frac{V_i^\star}{\epsilon} - 1 - \epsilon \log\left(e^{-1} \sum_{i=1}^{N_m} e^{\frac{V_i^\star}{\epsilon}} \right) \right)}, \tag{9}$$

where $V_i^\star = x(t_0)^{\mathrm{T}} P_i x(t_0)$ is known to be the optimal cost associated with each controller [4], with $\epsilon \in \mathbb{R}^+$ denoting the weight on unpredictability. The following theorem determines sufficient conditions of stability of the system under proactive switching [3].

Theorem 1. The switched system (1) with active controller gain K_i given by (7) has an asymptotically stable equilibrium point for every switching sequence if the average dwell time in each mode is bounded by,

$$\tau_D > \frac{\log\left(\max_{q,p \in \{1,\ldots,\text{card}(B_c)\}} \frac{\bar{\lambda}(P_p)}{\underline{\lambda}(P_q)} \right)}{\min_{p \in \{1,\ldots,\text{card}(B_c)\}} \frac{\underline{\lambda}(Q_p + P_p B_p R_p^{-1} B_p^{\mathrm{T}} P_p)}{\underline{\lambda}(P_p)}},$$

with an arbitrary chatter bound $S_0 > 0$.

Proof. The proof is presented in [3]. ∎

2) AS Detection: The AS detection policy operates according to the following condition [3].

$$\|e(t)\| \geqslant e_{i,\text{thres}}(t), \tag{10}$$

where $e_{i,\text{thres}}$ are thresholds for each mode given as,

$$e_{i,\text{thres}}(t) = 2\|\bar{w}\| \int_{t-\delta t}^{t} \|R_i u_i^\star(\tau)\| \mathrm{d}\tau + \bar{\lambda}(R_i)\|\bar{w}\|^2, \tag{11}$$

where $\bar{w}$ is the acceptable level of variation/noise in the detection level. The signal $e(t)$ is given by,

$$\begin{aligned} e(t) &= \hat{V}_i(x(t-\delta t)) - \hat{V}_i(x(t)) \\ &- \int_{t-T}^{t} (x(t)^{\mathrm{T}} Q_i x(t) + u_i^{\star\mathrm{T}}(t) R_i u_i^\star(t)) \mathrm{d}\tau. \end{aligned} \tag{12}$$

For the AS Detection effects, the policy is only functional when the AS is in control of the system in order to provide the human with some opportunity to detect attacks.

Then, assuming AS is in control and detects a potential attack via (10), an alarm for the current mode is set as,

$$a_{s(t)}(t) = \begin{cases} T_{\text{out}}, & \text{if (10)} \wedge a_{s(t)}(t) \leqslant 0, \\ a_{s(t)}(t), & \text{otherwise}, \end{cases} \tag{13}$$

where $T_{\text{out}} \in \mathbb{R}^+$ is a constant timeout specified for the attacked mode and $a_{s(t)}(t)$ is the current mode's alarm, which is explained in the following Reactive Switching subsection. During this timeout time, the mode cannot be used. In order to do so, the switching described earlier needs to be modified to consider 'reactive' switching (which again affects both AS and human control).

3) Reactive Switching: The reactive aspect of switching is quantified by,

$$a(t) = \{a_i(t) \mid i \in \{1, \ldots, N_m\}\},$$

where $a_i(t) \leqslant 0$ implies no attack detected (so mode is available to be switched to) while $a_i(t) > 0$ implies mode cannot be switched to due to detected attack.

Furthermore, the proactive-reactive switching takes place at the specified interval T. At the end of this interval, prior to conducting the switching itself, the alarm vector is adjusted for elapsed time after the timeout as,

$$a_i(t) = a_i(t - \delta t) - T(a_i(t - \delta t) > 0). \tag{14}$$

Thus, every mode with $a_i(t) > 0$ will be ignored in the mode switching until its respective timeout is complete (which eventually results in $a_i(t) \leqslant 0$).

4) Human Control and Detection: For the human control, a reference signal tracker control system was designed. Assuming human is controlling system (by way of the tool), the optimal control $u_i^\star(t)$ is described as [10],

$$u_i^\star(t) = G_i x_{\text{ref}}(t) - K_i x(t), \tag{15}$$

$$G_i = -(C(A - B_i K_i)^{-1} B_i)^{-1}, \ \forall i \in \{1, \ldots, N_m\}. \tag{16}$$

Unlike the AS detection, that is only considered whenever AS is in control of the system, the human detection works at all times and is given by,

$$a_{s(t)}(t) = \begin{cases} T_{\text{out}}, \ d_H(t) > 0 \wedge u_t(t) > 0 \wedge a_{s(t)}(t - \delta t) \leqslant 0, \\ a_{s(t)}(t - \delta t), \text{otherwise}, \end{cases}$$

where $d_H(t)$ is the human detection signal that is set to a value greater than 0 whenever the human presses the 'Detect Attack' button in the tool.

5) Including Attacks in Control: The attacks specified by (5) and (6) are finally included in the control vector $u(t)$ as,

$$u(t) = \begin{cases} u_i^\star(t), & u_t(t) = 0, \\ u_a(t), & u_t(t) = 1, \\ u_i^\star(t) + u_a(t), & u_t(t) = 2, \end{cases} \tag{17}$$

where $u_i^\star(t)$ depends on human or AS control as previously specified.

D. Trust Evaluation (Step 4)

Much of the trust metric was taken from [11], but with a few modifications. Following directly from [11], there exist two metrics for the 'mutual' trust, by means of human's trust on AS ($T_{H\to A}$) and AS trust on human ($T_{A\to H}$), given by,

$$\begin{aligned} T_{H\to A}(t) &= A_1 T_{H\to A}(t-T) + B_1 P_A(t) - B_2 P_A(t-T) \\ &\quad + F_s(t)(D_1 F_A(t) - D_2 F_A(t-T)), \end{aligned} \tag{18}$$

$$\begin{aligned} T_{A\to H}(t) &= A_2 T_{A\to H}(t-T) + C_1 P_H(t) - C_2 P_H(t-T) \\ &\quad + F_s(t)(E_1 F_H(t) - E_2 F_H(t-T)), \end{aligned} \tag{19}$$

where $A_1, A_2, B_1, B_2, C_1, C_2, D_1, D_2, E_1, E_2$ are constant coefficients while P_A and P_H are the performance of the AS and human respectively. Unlike [11], the fault rates F_A and F_H indicate the error rates of AS and human on detecting attacks respectively. Additionally, it must be noted that $T_{H\to A}$ and $T_{A\to H}$ are updated every T seconds, in accordance with the tool's update rate and $F_s(t)$ will be discussed later along with the fault rates.

Then, $P_A(t)$ and $P_H(t)$ are defined as,

$$P_A(t) = \begin{cases} (1-k_A)P_A(t-T) + k_A P_{A_{max}}, & h_c(t) \leqslant 0, \\ (1-k_H)P_A(t-T) + k_H P_{A_{min}}, & h_c(t) > 0, \end{cases} \tag{20}$$

$$h_c(t) = \begin{cases} 0, & \text{implies system under AS control,} \\ 1, & \text{implies system under human control,} \end{cases}$$

where $h_c(t)$ informs on which entity is controlling the system. Furthermore, $k_A, k_H \in (0,1)$ are performance coefficients for autonomous and manual mode respectively, and $P_{A_{\min}}, P_{A_{\max}} \in [0,1]$ are the minimum and maximum performance of the AS. In general, $P_A(t)$ increases near to $P_{A_{\max}}$ when AS is in control and drops close to $P_{A_{\min}}$ when human is in control. This is also seen in Figure 2 where for the most recent 33 seconds, the AS is in control and its $T_{H\to A}$ increases, while the opposite happens when human is in control in the most recent 50-33 seconds. Here, and through out the paper, it is noted that $A_1{=}A_2{=}B_1{=}B_2{=}C_1{=}C_2{=}E_1{=}E_2{=}1$, $D_1{=}D_2{=}0.75$ with $k_A{=}0.075$, $k_H{=}0.1$ and $P_{A_{\max}}{=}1$, $P_{A_{min}}{=}0$.

Then, the $P_H(t)$ follows exactly from [11] as

$$P_H(t){=}(P_{H_{\max}}{-}P_{H_{\min}}) \left(\frac{r(t)}{\beta}\right)^{\beta} \left(\frac{1{-}r(t)}{1{-}\beta}\right)^{1-\beta} {+}P_{H_{\min}}, \tag{21}$$

where $\beta \in (0,1)$ is the difficulty of the task, $P_{H_{\min}}$ and $P_{H_{\max}} \in [0,1]$ are human performance limits and $r(t)$ is given by,

$$r(t) = \left(1 - \frac{1}{\tau}\right) r(t-T) + \frac{(h_c(t) > 0)}{\tau}, \tag{22}$$

where $\tau > 0$ is the time constant which evaluates the effect of past utilization by human.

The overall effect of $P_H(t)$ can be seen again in Figure 2, where only the performance affects trust (no attacks). The overall pattern of $T_{A\to H}$ is different to $T_{H\to A}$ in that it rises immediately and falls back to ≈ 0 regardless of if human or AS is in control, which is due to β set to 0.3, which implies human performance is best around 30% of the entire time-sharing between human and AS, and drops elsewhere. Here, and throughout the paper it is noted that $\tau{=}5$, $P_{H_{\max}}{=}1$ and $P_{H_{\min}}{=}0$.

Given the nature of the two performance metrics, the trust limits (seen as black lines in Figure 2) were implemented different to [11], especially with respect to $T_{A\to H}$. These limits evaluated when the system should switch between human and AS, with a good example being the 0 second mark on the "Trust Performance" Graph in Figure 2, where the bright red "Suggested Action" warning on the left side of the UI asks human to take control of the system, set off by $T_{A\to H}$ dropping below the lower limit.

Specifically, there exist two types of limits: Overtrust and Undertrust, which occur when either human or AS take too much / little control of the system respectively. Formally, the undertrust and overtrust, via a 'switch control' variable $s_c(t)$, are set as,

$$s_c(t) = \begin{cases} 1, & T_{H\to A}(t) > T_{H\to A,u} \\ & \vee\, (h_c(t) \leqslant 0 \wedge T_{A\to H}(t) < T_{A\to H,l_A}), \\ -1, & T_{H\to A}(t) < T_{H\to A,l} \\ & \vee\, (h_c(t) > 0 \wedge T_{A\to H}(t) < T_{A\to H,l_H}), \\ 0, & \text{otherwise,} \end{cases} \tag{23}$$

where $s_c(t){=}1$ and $s_c(t){=}-1$ imply setting off a warning in the UI, as seen under "Suggested Action" in Figure 2, to switch to human and AS control respectively, while $s_c(t){=}0$ implies no change required (no warnings set off). Furthermore, the limits for $T_{H\to A}$ are $[T_{H\to A,l}, T_{H\to A,u}]$ and for $T_{A\to H}$ are $[T_{A\to H,l_A}, T_{A\to H,l_H}]$ respectively. Following [11], $T_{H\to A}$ has one upper and lower limit, where $T_{H\to A}(t) > T_{H\to A,u}$ in (23) implies human overtrusting AS (human needs to take control) and $T_{H\to A}(t) < T_{H\to A,l}$ implies human undertrusting AS (AS needs to take control). Similarly, $T_{A\to H}(t) < T_{A\to H,l_A}$ under AS control implies AS undertrusting human (human needs to take control) and $T_{A\to H}(t) < T_{A\to H,l_H}$ under human control implies AS overtrusting human (AS needs to take control). These limits can be seen in Figure 2 as $[T_{H\to A,l}, T_{H\to A,u}] = [0,1]$ and $[T_{A\to H,l_A}, T_{A\to H,l_H}] = [0,0]$.

The fault rates were then implemented based on detection of attacks. In accordance with how the limits are defined above, $F_s(t)$ from (18), (19) is defined as,

$$F_s(t) = \begin{cases} -1, & h_c(t) > 0, \\ 1, & h_c(t) \leqslant 0, \end{cases}$$

whose purpose is to flip the sign to ensure that the fault rates correctly affect the trust metrics based on who is in control. Then for the fault rates themselves, the following AS fault

Fig. 2. Tool in Action, simulated with no attacks

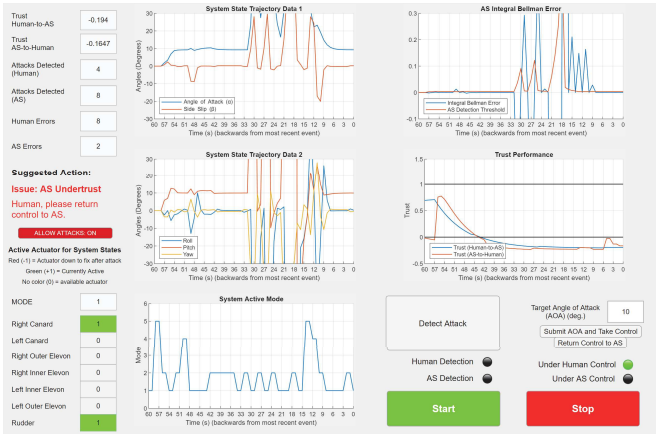

Fig. 3. Tool in Action with human controlling system to maintain 10 degrees of Angle of Attack (AOA), simulated with attacks (which shows both human errors and detections, as the state graphs constantly fluctuate due to the attacks)

$f_{d,A}(t)$ and human detection fault $f_{d,H}(t)$ are defined as,

$$f_{d,A}(t) = \begin{cases} 0, & h_c(t) > 0 \vee (u_t(t) \leqslant 0 \wedge a_{s(t)}(t) \leqslant 0), \\ -1, & u_t(t) > 0 \wedge a_{s(t)}(t) > 0, \\ 1, & u_t(t) > 0 \veebar a_{s(t)}(t) > 0, \end{cases}$$

$$f_{d,H}(t) = \begin{cases} -1, & u_t(t) > 0 \wedge d_H(t) > 0, \\ 1, & u_t(t) \leqslant 0 \wedge d_H(t) > 0, \\ 0, & \text{otherwise}, \end{cases}$$

where a $+1$ fault implies that the AS/Human failed at correctly detecting the attack and a -1 fault implies a correct detection. The fault rates were then determined via a moving average method. Since the method is the exact same for both human and AS failure rates, without loss of generality we only demonstrate it with AS fault rate. The moving average up to time t is defined via a set of 'windows,' each with fixed time T_w, average $w_{A_k}(t)$, and weighting $e_{A_k}(t) = \exp((k - w_n(t))/c)$ where $c > 0$ is a constant to scale the effect of the weighting. For the purposes of this simulation, $c = 6$ was used. Then $w_{A_k}(t)$ is defined as,

$$w_{A_k}(t) = \begin{cases} \frac{w_{A_k}(t-T)(\sigma(t)-1)+f_{d,A}(t)}{\sigma(t)}, & 0 \leqslant \frac{t}{kT_w} < 1, \\ w_{A_k}(t-T), & \text{otherwise}, \end{cases} \tag{24}$$

$$\text{where, } \sigma(t) = \frac{t \bmod T_w}{T} + 1.$$

The AS fault rate is then evaluated as,

$$F_A(t) = \frac{\sum_{k=1}^{w_n(t)} w_{A_k}(t) e_{A_k}(t)}{\sum_{k=1}^{w_n(t)} e_{A_k}(t)}. \tag{25}$$

It is noted again that the human fault rate $F_H(t)$ is found in the exact same way, except with using $f_{d,H(t)}$ in (24).

Lastly, Figure 3 can be observed for the effect of attacks while in human control, using target AOA of 10 degrees, and corresponding correct and incorrect human detections. The observed state fluctuations are due to the human analyst's inability to detect the attacks while the AS is inactive. Furthermore, the user clicked the "Detect Attack" button at 12 seconds, after which the attacks reduced, and a slight increase in human trust (upon making the correct detection) was seen.

IV. Tool Implementation

The UI of the tool itself was implemented in MATLAB App Designer, with the underlying algorithms implemented in MATLAB.

Algorithm 1: Main Algorithm
(called every T seconds from tool UI)

procedure
  Given an initial state $x(t_0)$, interval time T and system $(A, \{B_i\}, C)$.
  for $t = t + \delta t, t + 2\delta t \ldots, t + n\delta t = t + T$
    $\{u_a(t), u_t(t)\}$ = Algorithm 2.
    Read 'Target Angle of Attack' UI field into $x_{\text{ref}}(t)$, if $h_c(t) > 0$
    $\{u(t), s(t), a_{s(t)}(t), B_{s(t)}\}$ = Algorithm 3.
    Propagate Dynamics with Eq.(1).
    $\{T_{H\to A}(t), T_{A\to H}(t), s_c(t)\}$ = Algorithm 4.
  end for
end procedure

Algorithm 2: Attack Generator

Input: $s(t - \delta t), N_{\text{lim}}$.
Result: $u_a(t), u_t(t)$.
procedure
  Given $N_m, t_{\text{Dwell}}(t - T), t_{\text{Rest}}(t - T)$ and some N_a.
  if $t \bmod T \geqslant 0$ and $t \bmod T < \delta t$
    for $i = 1, \ldots, N_m$
      $t_{D_i}(t) \xleftarrow{\text{Eq.(2)}} t_{D_i}(t - T)$.
      $t_{R_i}(t) \xleftarrow{\text{Eq.(3)}} t_{R_i}(t - T)$.
    end for
  end if
  $u_a(t) \xleftarrow{\text{Eq.(5)}} t_{D_{s(t-\delta t)}}(t)$.
  $u_t(t) \xleftarrow{\text{Eq.(6)}} t_{D_{s(t-\delta t)}}(t)$.

return $u_a(t), u_t(t)$.
end procedure

Algorithm 3: Control System

Input: $A, \{B_i\}, C, x(t), x_{\text{ref}}(t), u_a(t), u_t(t), h_c(t), s(t)$.
Result: $u(t), s(t), a_{s(t)}(t), B_{s(t)}$.
procedure
  Given Q_i, R_i for all modes.
  if $t == t_0$ ▷ Initialize at time t_0
    Find all permutations (modes) of actuators and derive the subset of controllable pairs (A, B_i) of B_c.
    for $i = 1, \ldots, \text{card}(B_c) = N_m$
      $P_i \xleftarrow{\text{Eq.(8)}} \{A, B_i, Q_i, R_i\}$.
      $K_i \leftarrow \{R_i, B_i, P_i\}$.
      $p_i \xleftarrow{\text{Eq.(9)}} \{x(t_0), P_i\}$.
      $G_i \xleftarrow{\text{Eq.(16)}} \{A, B_i, K_i\}$.
    end for
  end if
  if $t \bmod T \geqslant 0$ and $t \bmod T < \delta t$ ▷ Proactive and Reactive Switching
    for $i = 1, \ldots, \text{card}(B_c) = N_m$
      $a_i(t) \xleftarrow{\text{Eq.(14)}} a_i(t - \delta t)$.
    end for
    Pick according to p_i an available mode.
  else
    $a_i(t) = a_i(t - \delta t)$.
  end if
  if $h_c(t) > 0$ ▷ Human Control
    $u^{\star}_{s(t)}(t) \xleftarrow{\text{Eq.(15)}} \{G_i, K_i, x(t), x_{\text{ref}}(t) | i = s(t)\}$.
  else ▷ AS Control and Detection
    $u^{\star}_{s(t)}(t) = -K_{s(t)} x(t)$.
    $e(t) \xleftarrow{\text{Eq.(12)}} \{Q_i, R_i, x(t), u^{\star}_{s(t)}(t), V_i^{\star}(t - \delta t), V_i^{\star}(t) | i = s(t)\}$.
    $e_{s(t),\text{thres}}(t) \xleftarrow{\text{Eq.(11)}} \{R_i, u^{\star}_{s(t)}(t), \bar{w}, \bar{\lambda} | i = s(t)\}$.
    $a_{s(t)}(t) \xleftarrow{\text{Eq.(13)}} a_{s(t)}(t)$.
  end if
  $u(t) \xleftarrow{\text{Eq.(17)}} \{u^{\star}_{s(t)}(t), u_a(t), u_t(t)\}$.
  return $s(t), u(t), a_{s(t)}(t), B_{i=s(t)}$.
end procedure

Algorithm 4: Trust Evaluation

Input: $u_t(t), a_{s(t)}(t), h_c(t), d_H(t)$.
Result: $T_{H\to A}(t), T_{A\to H}(t), s_c(t)$.
procedure
  Given constants required $(A1, A2, B1, \ldots, \beta, \tau)$ from (18) - (22) and trust limits $T_{H\to A,l}, T_{H\to A,u}, T_{A\to H,l_A}$, $T_{A\to H,l_H}$.
  Given $T_{H\to A}(t_0), T_{A\to H}(t_0), P_A(t_0), P_H(t_0), r(t_0), F_A(t_0), F_H(t_0), \{w_{A_k}(t_0)\}, \{e_{A_k}(t_0)\}, \{w_{H_k}(t_0)\}, \{e_{H_k}(t_0)\}$
  if $t \bmod T \geqslant 0$ and $t \bmod T < \delta t$
    $k_t = t$.
    $P_A(t) \xleftarrow{\text{Eq.(20)}} P_A(t - T)$.
    $P_H(t) \xleftarrow{\text{Eq.(21)}} \{P_H(t - T), r(t - T), h_c(t)\}$.
    $F_A(t) \xleftarrow{\text{Eq.(25)}} \{\{w_{A_k}(t_0)\}, \{e_{A_k}(t_0)\}\}$.
    $F_H(t) \xleftarrow{\text{Eq.(25)}} \{\{w_{H_k}(t_0)\}, \{e_{H_k}(t_0)\}\}$.
    $T_{H\to A}(t) \xleftarrow{\text{Eq.(18)}} \{T_{H\to A}(t - T), P_A(t), P_A(t - T), F_A(t), F_A(t - T), F_s(t)\}$.
    $T_{A\to H}(t) \xleftarrow{\text{Eq.(19)}} \{T_{A\to H}(t - T), P_H(t), P_H(t - T), F_H(t), F_H(t - T), F_s(t)\}$.
    $s_c(t) \xleftarrow{\text{Eq.(23)}} \{T_{H\to A}(t), T_{A\to H}(t), h_c(t)\}$.
    return $T_{H\to A}(t), T_{A\to H}(t), s_c(t)$.
  else
    return $T_{H\to A}(k_t), T_{A\to H}(k_t), s_c(k_t)$.
  end if
end procedure

V. Conclusion and Future Work

This work presented a fully realized proactive and reactive defense tool for CPS analyst augmentation and training, based on the principles of MTD and of Bellman-based detection mechanisms. We explored in detail the various routines of the software, both in their theoretical foundation and in their implementation. Finally, trust metrics were proposed in order to fully integrate this autonomous security tool with the human security analyst in a dynamic, intelligent fashion. Future efforts will focus on implementing the tool in actual CPS and showcasing its efficacy with experimental data.

References

[1] A. A. Cardenas, S. Amin, and S. Sastry, "Secure control: Towards survivable cyber-physical systems," in *Distributed Computing Systems Workshops, 2008. ICDCS'08. 28th International Conference on*. IEEE, 2008, pp. 495–500.

[2] C. Peng, H. Sun, M. Yang, and Y.-L. Wang, "A survey on security communication and control for smart grids under malicious cyber attacks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 8, pp. 1554–1569, 2019.

[3] A. Kanellopoulos and K. G. Vamvoudakis, "A moving target defense control framework for cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 1029–1043, 2019.

[4] K. G. Vamvoudakis and N.-M. T. Kokolakis, "Synchronous reinforcement learning-based control for cognitive autonomy," *Foundations and Trends® in Systems and Control*, vol. 8, no. 1–2, 2020.

[5] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, *Moving target defense: creating asymmetric uncertainty for cyber threats*. Springer Science & Business Media, 2011, vol. 54.

[6] K. E. Oleson, D. R. Billings, V. Kocsis, J. Y. Chen, and P. A. Hancock, "Antecedents of trust in human-robot collaborations," in *2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*. IEEE, 2011, pp. 175–178.

[7] P. Kaniarasu, A. Steinfeld, M. Desai, and H. Yanco, "Potential measures for detecting trust changes," in *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2012, pp. 241–242.

[8] M. F. Mahani, L. Jiang, and Y. Wang, "A bayesian trust inference model for human-multi-robot teams," *International Journal of Social Robotics*, pp. 1–15, 2020.

[9] X. Yu and J. Jiang, "Hybrid fault-tolerant flight control system design against partial actuator failures," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 4, pp. 871–886, 2011.

[10] G. F. Franklin, J. D. Powell, A. Emami-Naeini, and J. D. Powell, *Feedback control of dynamic systems*. Prentice hall Upper Saddle River, NJ, 2002, vol. 4.

[11] X. Wang, "Co-design of dynamic real-time scheduling and cooperative control for human-agent collaboration systems based on mutual trust," 2015.