Adaptive Intrusion Detection of Malicious Unmanned Air Vehicles Using Behavior Rule Specifications

Robert Mitchell and Ing-Ray Chen
Department of Computer Science
Virginia Tech
{rrmitche, irchen}@vt.edu

***Abstract*—In this paper, we propose an adaptive specification-based intrusion detection system (IDS) for detecting malicious unmanned air vehicles (UAVs) in an airborne system in which continuity of operation is of the utmost importance. An IDS audits UAVs in a distributed system to determine if the UAVs are functioning normally or are operating under malicious attacks. We investigate the impact of reckless, random and opportunistic attacker behaviors (modes which many historical cyber attacks have used) on the effectiveness of our behavior rule-based UAV IDS (BRUIDS) which bases its audit on behavior rules to quickly assess the survivability of the UAV facing malicious attacks. Through a comparative analysis with the multi-agent system/ant-colony clustering model (MAS/ACCM), we demonstrate a high detection accuracy of BRUIDS for compliant performance. By adjusting the detection strength, BRUIDS can effectively trade higher false positives for lower false negatives to cope with more sophisticated random and opportunistic attackers to support ultra safe and secure UAV applications.**

***Index Terms*—Intrusion detection, Unmanned air vehicles, security.**

I. Introduction

Unmanned air vehicles (UAVs) comprise a large part of the warfighting capability of modern militaries. Also, they are emerging in civilian applications such as surveillance for law enforcement, situational awareness for emergency services, content for news outlets and data collection for researchers. While they pose the same risk as piloted aircraft, the operator is removed from the vehicle in time and space which calls for enhanced automated security systems to guarantee the safe operation. Intrusion detection systems (IDSs) are security appliances that review audit data to identify cyber attacks that jeopardize this safe operation. Our goal in this work is to provide a general framework for intrusion detection of malicious UAVs in an airborne system.

In this paper, we propose a specification-based IDS called Behavior Rule-based Unmanned Air Vehicle (UAV) Intrusion Detection System (BRUIDS). It requires minimal run time resources and is adaptive, i.e., it can adapt to the attacker type and environment changes by adjusting its intrusion detection strength dynamically, so as to satisfy the maximum false negative rate ($p_{\rm fn}$) requirement while minimizing the false positive rate ($p_{\rm fp}$). We demonstrate that BRUIDS can effectively trade false positive rate for true positive rate to cope with sophisticated random and opportunistic attackers to support ultra safe and secure UAV applications. Through a comparative analysis, we demonstrate a high detection accuracy of our UAV intrusion detection technique. Three major contributions of our paper are the behavior-rule based intrusion detection theory, the modeling and analysis of our behavior-rule based intrusion detection design with simulation validation and a list of rules that describes the healthy, uncompromised behavior of a UAV.

The focus on UAV electronics protection is not well reported in the academic literature, only [4], [14], [15], [25] have studied this topic. However, IDS is well assessed in cyber communications, and vehicle-based electronics protection is well known. Blasch et al. [4] proposed a warplanning situational awareness tool that classifies outsiders in the physical domain as friendly, neutral or belligerent. Specifically, it is not an IDS that identifies insider attackers in the cyber domain. Trafton and Pizzi [25] described the role of intrusion detection in airborne military applications. The authors proposed Joint Airborne Network Services Suite (JANSS) which provides a framework to integrate an airborne military network. Among other services, JANSS covers intrusion detection. However, neither concrete solutions nor true/false positive rates were reported. Lauf and Robinson [14], [15] investigated HybrIDS, an anomaly-based approach. HybrIDS comprises two intrusion detection methods: Maxima Detection System (MDS) and Cross-Correlative Detection System (CCDS). MDS detects single intruders using audit data after a short training phase and accomplishes a more in-depth training phase for CCDS. CCDS can detect cooperating intruders after the longer training phase provided by MDS. HybrIDS was evaluated using a metric called "pervasion" defined as the percentage of malicious nodes in the system. It was reported that intruders can be detected even with a 22% pervasion.

With the exception of [4], existing work [14], [15], [25] had no numerical data regarding the false negative rate $p_{\rm fn}$ (i.e., missing a malicious node) and the false positive rate $p_{\rm fp}$ (i.e., misidentifying a normal node as a malicious node). In this paper we report ($p_{\rm fn}$, $p_{\rm fp}$) UAV IDS data to analyze the tradeoff between $p_{\rm fn}$ and $p_{\rm fp}$ obtained as a result of applying our proposed adaptive UAV IDS techniques. When graphing a standard ROC, we use $1 - p_{\rm fn}$ versus $p_{\rm fp}$ with $1 - p_{\rm fn}$ corresponding to the "true positive rate" the literature refers to.

The rest of the paper is organized as follows: In Section

II, we discuss the system model, including the UAV reference model, the threat model and the attacker archetypes. In Section III, we describe our UAV intrusion detection design with the goal to minimize the false negative rate without compromising the false positive rate. In Section IV, we report numerical data. In Section V, we perform a comparative analysis with a multitrust-based IDS scheme and demonstrate the superiority of our design. In Section VI, we survey related work. In Section VIII, we conclude the paper and outline future work.

II. System Model

A. Reference UAV

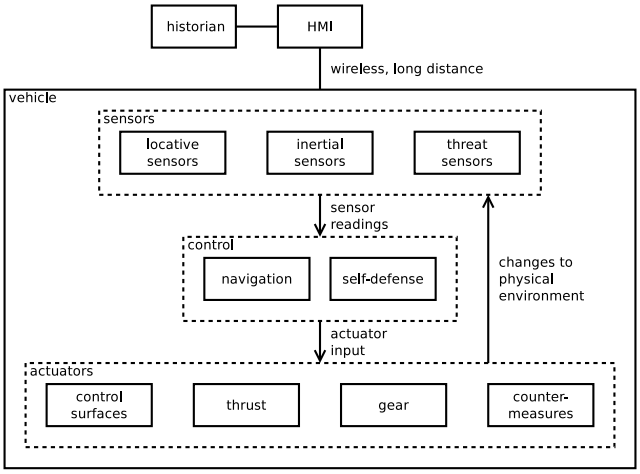

Fig. 1. Reference UAV Key Components.

We consider multiple UAVs each embedding cyber physical system (CPS) physical components (sensors and actuators). Figure 1 illustrates a reference UAV embedding physical sensors and actuators. One cyber physical loop in this model is the flight control system in each UAV. Inertial sensors drive realtime adjustment of control surfaces and thrust. In addition, locative sensors (navigational components), such as Global Positioning System (GPS), Global Navigation Satellite System (GLONASS), Compass, Galileo or inertial sensors drive non-realtime adjustment of control surfaces and thrust. Another cyber physical loop in this model is the threat countermeasures system. Radar components detect physical presence of threats, and specifically tuned radios detect radio frequency (RF) signatures of threats. These sensors drive the realtime deployment of countermeasures like flare, chaff and electronic countermeasures (ECM). Tian et al. [24] discussed major issues in jamming that are associated with the communications literature. On top of UAVs sit the historian and human machine interface (HMI) modules which may be replicated to provide UAV control functions over long distance; the literature also refers to this segment as a ground control station (GCS). For readability, we will use the terms “node” and “UAV” interchangeably in the paper. The UAV reference model accounts for some general behaviors of a UAV to allow us to quickly assess the survivability of the UAV facing malicious attacks.

B. Threat Model

It is important to define the threat model to cover system vulnerabilities since our UAV IDS technique is based on behavior rules specifying expected normal behaviors of a sensor or actuator embedded in a UAV for detection of these threats. We base this threat model on domain experience, literature review [13], [28] and current events [9].

We consider seven threats towards a UAV:

1) The first threat is an attacker that directs a UAV’s weapon against a friendly resource.
2) The second threat is an attacker that corrupts data the UAV reports.
3) The third threat is an attacker that promotes confederate UAVs and marginalizes legitimate UAVs.
4) The fourth threat is an attacker that captures a UAV by deploying the landing gear when outside of the air base.
5) The fifth threat is an attacker that exfiltrates mission data.
6) The sixth threat is an attacker that activates a UAV’s countermeasures unnecessarily.
7) The seventh threat is an attacker that decreases a UAV’s endurance by wasting its energy.

Threats 1-4 attack the integrity. Threat 5 attacks the confidentiality. Threats 6 and 7 attacks the availability.

C. Monitoring Techniques

Our behavior-rule based IDS approach relies on the use of monitor nodes. We assume that a monitor node performs intrusion detection on a trusted neighbor node. One possible design is to have a sensor (actuator) monitor another sensor (actuator respectively) within the same UAV. However, this design requires each sensor (actuator) to have multiple sensing functionalities. Another design which we adopt is to have a neighbor UAV or a remote HMI monitor a trusted UAV. We model imperfect monitoring by an error parameter, $p_{\rm err}$, representing the probability of a monitor node misidentifying the status of the trusted node due to ambient noise and/or wireless communication faults in airborne environments. In general a node may deduce $p_{\rm err}$ at runtime by sensing the amount of ambient noise and wireless communication errors around it. $p_{\rm err}$ is not fixed in a particular detection scenario. Rather, $p_{\rm err}$ spans a range of values to model a dynamically changing mis-monitoring error probability due to ambient noise and wireless communication faults reflecting dynamically changing environment conditions. Here we note that while $p_{\rm err}$ changes dynamically depending on environment conditions, it does not depend on the state a trusted UAV is in.

D. Attacker Archetypes

We differentiate three attacker archetypes: reckless, random and opportunistic. A reckless attacker performs attacks whenever it has a chance. The main objective is to impair the UAV functionality at the earliest possible time. A random attacker, on the other hand, performs attacks only randomly to avoid detection. It is thus insidious and deceptive with the objective

to cripple the UAV functionality. We model the attacker behavior by a random attack probability p_a. When $p_a = 1$ the attacker is a reckless adversary. An opportunistic attacker is the third archetype we consider. It exploits the environment noise modeled by $p_{\rm err}$ (probability of mis-monitoring) to perform attacks. While a random attacker's p_a is fixed, an opportunistic attacker decides its attack probability p_a based on $p_{\rm err}$ sensed. When $p_{\rm err}$ is higher, the system is more vulnerable, so its p_a is higher. An opportunistic attacker can be conservative or aggressive. We apply the demand-pricing relation function, i.e., $\text{Demand} = C \times \text{Pricing}^{-\varepsilon}$ in the field of Economics [1], [8], [29], which describes how demand changes when pricing changes, to model the relation between the opportunistic attackers attack probability p_a (mapped to demand) and the imperfect monitoring probability $p_{\rm err}$ (mapped to pricing). The demand-pricing relation function predicts that demand shall decrease when pricing increases and vice versa, the degree of which is controlled by the elastic constant ε which determines the effect of pricing change. Because both p_a and $p_{\rm err}$ are real numbers between 0 and 1, let:

$$p_a = \mathcal{C} \times p_{\rm err}^{\varepsilon}. \tag{1}$$

where $\mathcal{C} > 0$ covers both conservative and aggressive attack behaviors:

1) $\varepsilon = 1$: p_a increases linearly with $p_{\rm err}$; this models a conservative opportunistic attacker.
2) $\varepsilon < 1$: p_a increases exponentially with $p_{\rm err}$; this models an aggressive opportunistic attacker, the extent of which is modeled by ε.

III. UAV Intrusion Detection Design

A. Behavior Rules

Our IDS design for the reference UAV model relies on the use of simple specification-based *behavior rules* for each UAV. They are oriented toward detecting an inside attacker attached to embedded sensors or actuators, provide a continuous output between 0 and 1, and allow a monitor node to perform intrusion detection on a trusted neighbor through monitoring. Table I lists the behavior rules for detecting a malicious UAV with the monitor being a peer UAV or an HMI (see Figure 1). Since behavior rules are derived from the threat model, Table I also lists the threat from which a behavior rule originates.

We prioritize the behavior rules so that more impactful rules are searched first. Table I organizes the behavior rules by decreasing impact. The highest priority behavior rules protect integrity, the next priority behavior rules protect confidentiality, the third priority behavior rules concern availability. The behavior rules are optimally searched with the priority criteria.

TABLE I
UAV Behavior Rules

Threat Index	Description	Priority Criteria
1	safe weapons if outside battlespace	integrity
2	produce accurate data	integrity
3	provide true recommendations	integrity
4	stow landing gear if outside domestic air base	integrity
5	do not send to non-whitelisted destinations	confidentiality
6	turn off countermeasures if no threat	availability
7	use minimum thrust if loitering	availability

*The trusted node is a UAV, and the monitor is a peer UAV or an HMI for all behavior rules.

B. Transforming Rules to State Machines

Each behavior rule does not specify just one state, but a number of states, some of which are safe states in which normal behavior (obedience of this behavior rule) is observed, while others are unsafe states in which malicious behavior (violation of this behavior rule) is observed. A behavior rule thus has a number of state variables, each with a range of values, together indicating whether the node is in normal or malicious behavior status (with respect to this rule).

The following procedure transforms a behavior specification into a state machine: First we identify the attack behavior indicator as a result of a behavior rule being violated. Then we transform this attack behavior indicator into a conjunctive normal form predicate and identify the involved state components in the underlying state machine. Next we combine the attack behavior indicators into a Boolean expression in disjunctive normal form. Then we transform the union of all predicate variables into the state components of a state machine and establish their corresponding ranges. Finally we manage the number of states by state collapsing and identifying combinations of values that are not legitimate.

Below we exemplify how a state machine is derived from the behavior specification in terms of behavior rules for the reference UAV model.

1) Identify Attack Behavior Indicators: Attacks performed by a compromised sensor (actuator) embedded in a UAV will drive the UAV into certain attack behavior indicators identifiable through analyzing the specification-based behavior rules. There are seven attack behavior indicators as a result of violating the seven behavior rules for a UAV listed in Table I.

The first UAV attack behavior indicator is that a UAV readies its weapon when outside the battlespace (when within its domestic air base or coalition air corridor). This indicator catches attackers that intend to direct a UAV's weapon against a friendly resource; these attackers attach to the UAV weapon module. The second UAV attack behavior indicator is that a trusted node's embedded sensor reading differs from the monitor's embedded sensor reading. The monitor is in the neighborhood of the trusted node, measuring the same physical phenomenon. The third UAV attack behavior indicator is that a monitor UAV provides bad-mouthing attacks, i.e., providing bad recommendations regarding a well-behaving trusted UAV, or good-mouthing attacks, i.e., providing good recommendations regarding a misbehaving trusted UAV. This is detected by comparing recommendations provided by multiple monitor UAVs and detecting discrepancies. The fourth UAV attack behavior indicator is that a UAV deploys landing gear when outside its domestic air base. This indicator catches attackers that intend to capture the UAV; these attackers control the UAV

TABLE II
UAV ATTACK BEHAVIOR INDICATORS IN CONJUNCTIVE NORMAL FORM

Attack Behavior Indicator	Expression
1	(Weapons = READY) $\wedge$ (Location $\neq$ BATTLESPACE)
2	$\mid$ Trusted Node Data - Monitor Data $\mid > \delta$
3	Trusted Node Audit $\neq$ Monitor Audit
4	(Gear = DEPLOYED) $\wedge$ (Location $\neq$ DOMESTIC AIR BASE)
5	Destination $\neq$ WHITELISTED
6	(Countermeasures = ACTIVE) $\wedge$ (Threat = FALSE)
7	(Thrust $>$ T) $\wedge$ (Status = LOITER)

landing gear module. One way an attacker could pursue this goal is to launch a shellcode attack that diverts the UAV to an area they control. The fifth UAV attack behavior indicator is that a node sends bytes to unauthorized parties. Explicitly authorized parties are said to be "whitelisted." This indicator catches attackers that intend to exfiltrate mission data. The sixth UAV attack behavior indicator is that a UAV uses countermeasures without identifying a threat. This indicator catches attackers that intend to increase the vulnerability and decrease the availability of a UAV; these attackers attach to the UAV countermeasures module. The seventh UAV attack behavior indicator is that a loitering UAV uses more than the minimum thrust required to maintain altitude. This indicator catches attackers that intend to decrease a UAV's endurance by wasting its energy; these attackers attach to the UAV thrust module.

2) Express Attack Behavior Indicators in Conjunctive Normal Form: Table II lists the UAV attack behavior indicators in Conjunctive Normal Form. Here we note that each attack behavior indicator may have several state variables. For example, attack behavior indicator 1 in Table II has two state variables (or components), namely, Weapons and Location, in the underlying state machine.

3) Consolidate Predicates in Disjunctive Normal Form: ((Weapons = READY) $\wedge$ (Location $\neq$ BATTLESPACE)) $\vee$ ($|$Trusted Node Data - Monitor Data$|$ $>$ δ) $\vee$ (Trusted Node Audit $\neq$ Monitor Audit) $\vee$ ((Gear = DEPLOYED) $\wedge$ (Location $\neq$ DOMESTIC AIR BASE)) $\vee$ (Destination $\neq$ WHITELISTED) $\vee$ ((Countermeasures = ACTIVE) $\wedge$ (Threat = FALSE)) $\vee$ ((Thrust $>$ T) $\wedge$ (Status = LOITER))

4) Identify State Components and Component Ranges: We limit the range of the altitude parameter to the lowest point on Earth and maximum altitude of a large UAV.

We quantize continuous components at integer scale in permissible ranges. For example, altitude is in the range of $[-423\text{ m}, 15000\text{ m}]$ and bank is in the range of $[-180°, 180°]$. Table III shows a complete list of the permissible ranges of UAV state components. The resulting HMI and UAV automatons have $181\times361\times15424\times361\times361\times361\times2\times2\times2\times2\times101\times201\times201\times201\times2\times2\times2\times2\times2 = 9.9553\times10^{27}$ states. Both of these automata are too large; we deal with this state explosion in the next step.

5) Manage State Space: Reducing the size of the state machine is an important subproblem for this study. Rather than approaching state machine reduction in an hoc fashion,

TABLE III
UAV STATE COMPONENTS

Name	Control or Reading	Range
latitude	reading	$[-90°, 90°]$
longitude	reading	$[-180°, 180°]$
altitude	reading	$[-423\text{ m}, 15000\text{ m}]$
bank	reading	$[-180°, 180°]$
pitch	reading	$[-180°, 180°]$
yaw	reading	$[-180°, 180°]$
threat	reading	true, false
stall	reading	true, false
data	reading	match, mismatch
audit	reading	match, mismatch
thrust	control	$[0, 100\%]$
aileron	control	$[100\%$ left, 100% right$]$
elevator	control	$[100\%$ down, 100% up$]$
rudder	control	$[100\%$ left, 100% right$]$
gear	control	deployed, stowed
countermeasure	control	active, inactive
weapon	control	safe, ready
destination	control	whitelisted, unauthorized

we applied this broad strategy:

1) identify maximum acceptable state machine size;
2) while the state machine is too large:
   a) choose the state component with the largest domain;
   b) compress the domain of this state component.

To manage the number of states, we reduce the size of the state machine by abbreviating the values for and consolidating some components. Our rules only consider three values for position (domestic air base, coalition air corridor or battlespace), so we consolidate latitude, longitude and altitude into a single position component and restrict this component to three values. Our rules only consider four values for flight status (takeoff, travel, loiter or land), so we consolidate bank, pitch, yaw, stall, aileron, elevator and rudder into a single status component and restrict this component to four values. Our rules only consider three values for thrust (sub-stall, minimal or super-minimal) so we restrict this component to three values. This treatment yields a modest state machine with $3\times4\times2\times2\times2\times3\times2\times2\times2\times2 = 4608$ states, out of which 165 are identified as safe states and 4443 are unsafe states.

While this flat approach to reducing the state machine met our needs, results with other configurations may vary. If this strategy does not yield a reasonably sized automaton, hierarchical state analysis is an orthogonal approach that will further manage the state space.

6) Behavior Rule State Machine: For the UAV state machine, there are 165 safe states and 4443 unsafe states. First we label these states as states $1, 2, \ldots, n = 4608$. Next we decide p_{ij}, the probability that state i goes to state j, for each (i, j) pair in the state machine to reflect a normal (or malicious) UAV's behavior.

For a compromised UAV, p_{ij} depends on its attacker type: A reckless attacker will not go from an unsafe state to a safe state because it continuously attacks. So $p_{ij} = 0$ if i is an unsafe state and j is a safe state. However, the monitoring process is imperfect with error probability p_{err}, so the monitor node may not observe exactly the same state transitions performed

by the reckless attacker. As a result, $p_{ij} = p_{\text{err}}$ instead of $p_{ij} = 0$ when i is an unsafe state and j is a safe state.

For a random attacker with attack probability p_a, p_{ij} values sum to p_a for a given i for all unsafe states j and p_{ij} values sum to $1 - p_a$ for a given i for all safe states j because it will stop attacking with probability $1 - p_a$. With imperfect monitoring, a monitor node sees: p_{ij} values sum to $p_a \times (1 - p_{\text{err}}) + (1 - p_a) \times p_{\text{err}}$ for a given i for all unsafe states j and p_{ij} values sum to $(1 - p_a) \times (1 - p_{\text{err}}) + p_a \times p_{\text{err}}$ for a given i for all safe states j.

In practice, during the testing phase one will seed an attacker and assign a monitor node to observe the states this attacker enters to assign individual p_{ij} values. For the special case in which every unsafe state among all is entered with equal probability and every safe state among all is also entered with equal probability, a compromised UAV with random attack probability p_a will be observed as having p_{ij} as $((1 - p_a) \times (1 - p_{\text{err}}) + p_a \times p_{\text{err}})/165$ when j is one of the 165 safe states, and as $(p_a \times (1 - p_{\text{err}}) + (1 - p_a) \times p_{\text{err}})/4443$ when j is one of the 4443 unsafe states. Figure 2 illustrates the behavior rule state machine for a compromised UAV. Crossed dotted slashes over a state indicate an unsafe state.

Here we note that the random attacker behavior-rule state machine derived above covers all three attacker models: reckless for which $p_a = 1$, random for which $p_a < 1$, and opportunistic for which p_a is related to p_{err} by Equation 1.

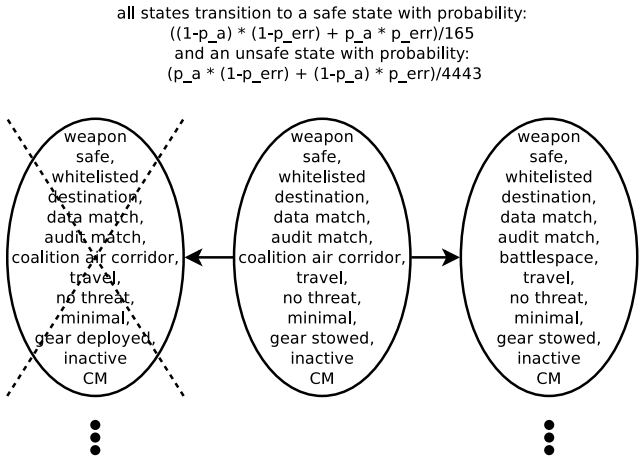

Fig. 2. Random Attacker Behavior Rule State Machine.

For a normal UAV, it should stay in safe states 100% of the time; however, occasionally it may be misidentified by the monitor node as staying in an unsafe state due to ambient noise, temporary system faults and wireless communication faults with error probability p_{err}. For the special case in which every unsafe state among all is entered with equal probability and every safe state among all is also entered with equal probability, a normal UAV node will be observed as having p_{ij} as $(1 - p_{\text{err}})/165$ when j is one of the 165 safe states, and as $p_{\text{err}}/4443$ when j is one of the 4443 unsafe states. Figure 3 illustrates the behavior rule state machine for a normal UAV. Again, crossed dotted slashes over a state indicate an unsafe state. While they look similar, Figures 2 and 3 are actually different because the transition rates are different for a random attacker (Figure 2) versus for a normal node (Figure 3). They look similar because they have the same number of states as generated from the behavior rules.

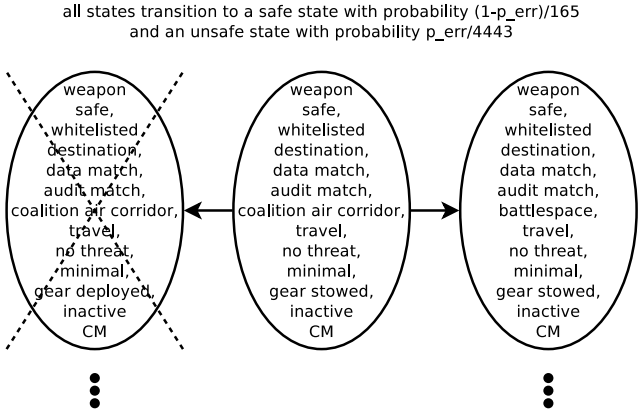

Fig. 3. Good Node Behavior Rule State Machine.

C. Collect Compliance Degree Data

Our BRUIDS relies on the use of monitor nodes, e.g., a UAV or HMI is a monitor node of another UAV. The monitor node knows the state machine of the trusted node assigned to it. The monitor node periodically measures the amount of time the trusted node stays in safe and unsafe states as the trusted node migrates from one state to another triggered by events causing state transitions. We consider a binary grading policy, i.e., assigning a compliance degree of 1 to a safe state and 0 to an unsafe state.

Let c be the compliance degree of a node. The compliance degree c of a node essentially is equal to the proportion of the time the node is in safe states. Let $\mathcal{S}$ be the set of safe states the trusted node traverses over a period of time T. Let t_i be the amount of time that the trusted node stays in a safe state i, as measured by the monitor node. Then the monitor node collects an instance of compliance degree c by:

$$c = \frac{\sum_{i \in \mathcal{S}} t_i}{T} \tag{2}$$

If a node stays only in safe states during T, then by Equation 2, its compliance degree c is one. On the other hand, if a node stays only in unsafe states only during T, then its compliance degree c is zero. The monitor node monitors and collects the trusted node's compliance degree history $c_1, c_2, \ldots, c_n$ for n monitoring periods, where n is sufficiently large, based on which it concludes whether or not the trusted node is compromised.

We leverage the state machines generated to collect compliance degree data of a normal UAV (or a malicious UAV) during the testing phase. Following Equation 2 which measures compliance degree as the proportion of time a trusted UAV is in safe states, the compliance degree c is essentially equal to the sum of the probabilities of safe states i.e., $c = \sum_{j \in \mathcal{S}} \pi_j$, where π_j is the limiting probability that the node is in state j of the state machine and $\mathcal{S}$ is the set of safe states in the

state machine. We utilize Monte Carlo simulation to obtain the limiting probability π_j. Specifically, we collect compliance degree history $c_1, c_2, \ldots, c_n$ of a UAV with n runs of Monte Carlo simulation. In run i, given a normal (or a malicious) UAV's state machine as input, we start from state 0 and then follow the stochastic process of this node as it goes from one state to another. We continue doing this until at least one state is reentered sufficiently often (say 100 times). Then we calculate π_j using the ratio of the number of transitions leading to state j to the total number of state transitions. After π_j is obtained, we collect c_i in the ith simulation run by $\sum_{j \in \mathcal{S}} \pi_j$. We repeat a sufficiently large n test runs to collect $c_1, c_2, \ldots, c_n$ needed for computing the distribution of the compliance degree of a normal UAV or a malicious UAV performing reckless, random or opportunistic attacks.

D. Compliance Degree Distribution

The measurement of compliance degree of a node frequently is not perfect and can be affected by noise and unreliable wireless communication in the airborne system. We model the compliance degree by a random variable X with $G(\cdot) = Beta(\alpha, \beta)$ distribution [20], with the value 0 indicating that the output is totally unacceptable (zero compliance) and 1 indicating the output is completely acceptable (perfect compliance), such that $G(a)$, $0 \leq a \leq 1$, is given by

$$G(a) = \int_0^a \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}\, dx \tag{3}$$

and the expected value of X is given by

$$E_B[X] = \int_0^1 x\, \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}\, dx = \frac{\alpha}{\alpha+\beta} \tag{4}$$

The α and β parameters are to be estimated based on the method of maximum likelihood by using the compliance degree history collected $(c_1, c_2, \ldots, c_n)$ during the system's testing phase. We choose the *Beta* distribution because it is defined on the interval $[0, 1]$, continuous and used across many disciplines.

We consider a single parameter $Beta(\beta)$ distribution with α equal to 1. In this case, the density is $\beta(1-x)^{\beta-1}$ for $0 \leq x \leq 1$ and 0 otherwise. The maximum likelihood estimate of β is

$$\hat{\beta} \;=\; \frac{n}{\sum_{i=1}^{n} \log(\frac{1}{1-c_i})} \tag{5}$$

E. False Positive and Negative Rates

Our intrusion detection is characterized by false negative and false positive rates, denoted by p_{fn} and p_{fp}, respectively. A false positive occurs when a normal UAV is misdiagnosed as malicious, while a false negative occurs when a malicious UAV is missed as normal. While neither is desirable, a false negative is especially impactful to the system's continuity of operation. While many detection criteria [2], [6], [7] are possible, we consider a threshold criterion in this paper. That is, if a malicious node's compliance degree denoted by X_b with a probability distribution obtained by Equation 3 is higher than a system minimum compliance threshold C_T then there is a false negative. Suppose that the compliance degree X_b of a malicious node is modeled by a $G(\cdot) = Beta(\alpha, \beta)$ distribution. Then the host IDS false negative rate p_{fn} is given by:

$$p_{\text{fn}} = \Pr\{X_b > C_T\} \;= 1 - G(C_T). \tag{6}$$

On the other hand, if a normal node's compliance degree denoted by X_g is less than C_T then there is a false positive. Again, suppose that the compliance degree X_g of a normal node is modeled by a $G(\cdot) = Beta(\alpha, \beta)$ distribution. Then the host false positive rate p_{fp} is given by:

$$p_{\text{fp}} = \Pr\{X_g \leq C_T\} = G(C_T). \tag{7}$$

TABLE IV
β IN BETA(1,β) AND RESULTING p_{fn} AND p_{fp} VALUES UNDER VARIOUS RANDOM ATTACK MODELS FOR UAV ($C_T = 0.90$, $p_{\text{err}} = 0.01$).

p_a	β	p_{fn}	p_{fp}
1.00	99.3	$< 0.001\%$	2.30%
0.80	4.33	0.005%	2.30%
0.40	1.10	8.02%	2.30%
0.20	0.632	23.3%	2.30%
0.10	0.449	35.5%	2.30%

TABLE V
β IN BETA(1,β) AND RESULTING p_{fn} AND p_{fp} VALUES UNDER VARIOUS OPPORTUNISTIC ATTACK MODELS FOR UAV ($C_T = 0.90$, $p_{\text{err}} = 0.01$, $\mathcal{C} = 10$).

Model	β	p_{fn}	p_{fp}	ε	p_a
aggressive	0.734	18.5%	2.30%	0.8	0.251
aggressive	0.555	27.9%	2.30%	0.9	0.158
conservative	0.449	35.5%	2.30%	1.0	0.1

IV. NUMERICAL DATA

We report numerical data in this section. We execute the procedure described in Section III to collect a sequence of compliance degree values $(c_1, c_2, \ldots, c_n)$ for $n = 1000$ Monte Carlo simulation runs for the UAV. We then apply Equation 5 to compute the β parameter value of $G(\cdot) = Beta(\alpha, \beta)$ for the probability distribution of the compliance degree for a normal node or a malicious node. We then calculate p_{fn} and p_{fp} by Equations 6 and 7, respectively. We adjust the minimum compliance threshold C_T to control p_{fn} and p_{fp} obtainable.

Tables IV and V exemplify the β values and the resulting p_{fn} and p_{fp} values obtained when C_T is 0.9 (C_T is a design parameter to be fine-tuned to trade high false positives for low false negatives) and $p_{\text{err}} = 0.01$ for random and opportunistic attackers, respectively.

Since $\alpha/(\alpha+\beta)$ (with $\alpha = 1$) is the expected compliance degree detected out of a trusted UAV, the β value detected is sensitive and adaptive to the attacker archetypes. A reckless attacker with $p_a = 1$ will reveal a low compliance degree and, hence, will have a high β value. A random attacker who performs attacks only probabilistically with rate p_a will have a relatively high compliance degree and, hence, a relatively low β value. This trend of increasing β with increasing p_a is clearly shown in Table IV. For opportunistic attackers, aggressive attackers tend to reveal a relatively low compliance degree compared with conservative attackers and, hence, a relatively

high β value. This trend of increasing β with increasing aggressiveness is demonstrated in Table V.

From Table IV, we observe that when the random attack probability p_a is high, the attacker can be easily detected as evidenced by a low false negative rate. Especially when $p_a = 1$, a reckless attacker can hardly be missed. On the other hand, as p_a decreases, a random attacker becomes more hidden and insidious and the false negative rate increases.

From Table V, we observe that the resulting p_{fn} values obtained depend on the aggressiveness of opportunistic attackers. More aggressive opportunistic attackers (those with a smaller ε) will have a better true positive rate $(1 - p_{\text{fn}})$ because of a higher attacker probability p_a being used.

Here we note that in both Tables IV and V, the false positive rate p_{fp} remains the same regardless of the random attack probability, because p_{fp} measures the probability of misidentifying a normal node only.

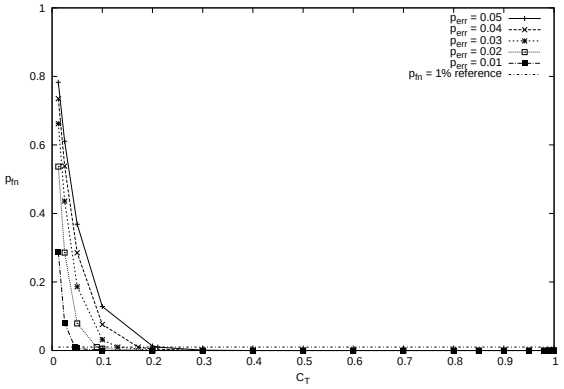

Fig. 4. False Negative Rate Versus Compliance Threshold for a Reckless Attacker under Varying p_{err}.

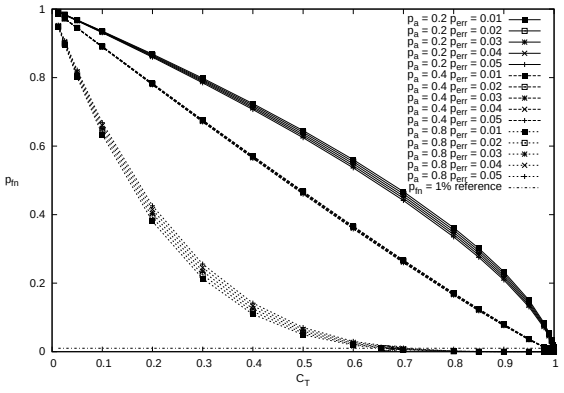

Fig. 5. False Negative Rate Versus Compliance Threshold for a Random Attacker under Varying p_{err}.

Figures 4, 5 and 6 illustrate the effect of C_T and p_{err} on p_{fn} of different types of attackers. They show the false negative rate decreases as C_T increases; this is because as the compliance threshold increases, a malicious node is less likely to evade detection. While Figure 4 shows that a larger p_{err} will benefit a reckless attacker, a smaller p_{err} will benefit random and opportunistic attackers with a small attack probability p_a.

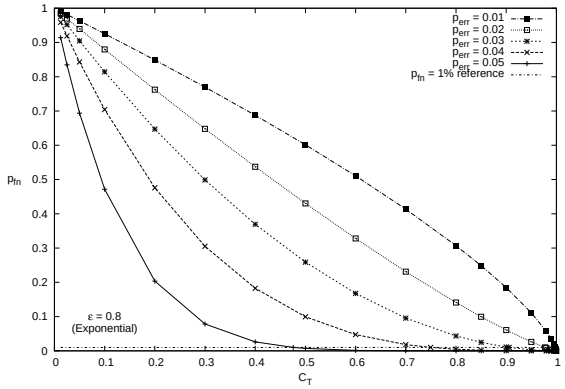

Fig. 6. False Negative Rate Versus Compliance Threshold for an Opportunistic Attacker under Varying p_{err}.

This is because p_{err} will obscure some of uniformly hostile behavior of the reckless attacker but will undermine the deceptively compliant behavior of the random and opportunistic attackers with a small p_a. Figure 5 shows that a smaller p_{err} will benefit random attackers when p_a is small (0.2), a larger p_{err} will benefit random attackers when p_a is large (0.8).

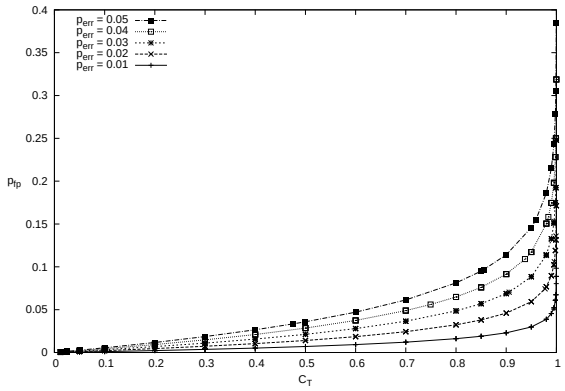

Fig. 7. False Positive Rate Versus Compliance Threshold for a Good Node under Varying p_{err}.

Figure 7 illustrates the effect of p_{err} on p_{fp} for a normal node. It shows the false positive rate increases as C_T increases; this is because as the compliance threshold increases, a normal node is more likely to be incorrectly misidentified as a malicious node. Also, it shows a smaller p_{err} will benefit a normal node; this is because p_{err} distorts the uniformly compliant behavior of a normal node.

The results obtained above can be used by the system to adaptively select the minimum C_T value dynamically to satisfy the imposed p_{fn} requirement while minimizing p_{fp} as much as possible in response to the environment condition (e.g., ambient noise) and the suspected attacker type detected at runtime (e.g., a random attacker). Table VI illustrates a scenario in which the maximum p_{fn} allowable is 1%, which must be satisfied. Given a p_{err} value and the attacker type as input, there is a C_T value at which $p_{\text{fn}} = 1\%$ as decided from Figures 4, 5, 6 (following the horizontal dashed line at $p_{\text{fn}} = 0.01$). Then from the C_T value selected, one can decide

TABLE VI
C_T TO SATISFY p_{fn} (1%) WHILE MAXIMIZING p_{fp} GIVEN p_{err} AND ATTACKER TYPE AS INPUT ($p_a = 0.2$, $\varepsilon = 0.8$)

	Reckless			Random			Opportunistic		
p_{err}	C_T	p_{fn}	p_{fp}	C_T	p_{fn}	p_{fp}	C_T	p_{fn}	p_{fp}
0.01	0.045313	0.01	0.000468	0.999315	0.01	0.070959	0.998123	0.01	0.061446
0.02	0.088882	0.01	0.001898	0.999219	0.01	0.135857	0.977295	0.01	0.074340
0.03	0.131062	0.01	0.004335	0.999109	0.01	0.195247	0.905752	0.01	0.070442
0.04	0.171620	0.01	0.007814	0.998992	0.01	0.249849	0.749324	0.01	0.056019
0.05	0.210615	0.01	0.012370	0.998862	0.01	0.300077	0.475645	0.01	0.033407

the resulting p_{fp} based on Equation 7. Table VI summarizes the C_T settings for all attacker types over a range of p_{err}. For example, the system manager should set C_T to 0.905752 when facing an opportunistic attacker with $p_{\text{err}} = 0.03$, and $\varepsilon = 0.8$ to achieve $p_{\text{fn}} = 1\%$ and $p_{\text{fp}} = 7\%$. This C_T value is obtained by following the middle curve in Figure 6 intersecting with the horizontal dashed line at $p_{\text{fn}} = 1\%$.

Table VI illustrates adaptive IDS design: the system selects the minimum C_T value that would satisfy the maximum p_{fn} requirement while providing a p_{fp} as small as possible, given knowledge of the attacker type and p_{err}. An an example, suppose a suspected attacker is of opportunistic type whose attacking behavior is characterized by $\varepsilon = 0.8$ as described in Table VI. After determining $p_{\text{err}} = 0.01$ (which is detectable), the system would select a high C_T value (i.e., $C_T = 0.998123$) to yield $p_{\text{fn}} = 1\%$ while minimizing p_{fp} to 6.1446%. If the attacker type is reckless, on the other hand, then the system would select a low C_T value (i.e., $C_T = 0.045313$) to yield $p_{\text{fn}} = 1\%$ while minimizing p_{fp} to 0.0468%.

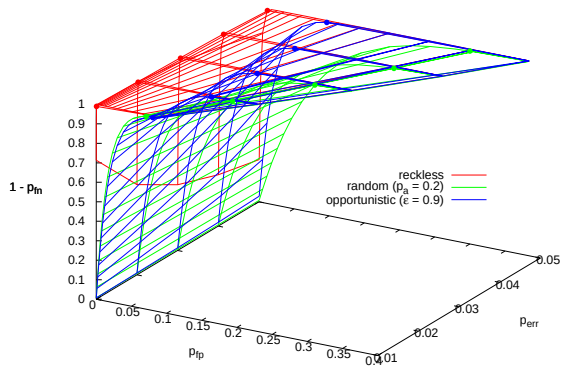

Fig. 8. BRUIDS Receiver Operating Characteristic Graph.

By adjusting C_T, our specification-based IDS technique can effectively trade higher false positives for lower false negatives to cope with more sophisticated and hidden random attackers. This is especially desirable for ultra safe and secure UAV applications for which a false negative may have a dire consequence. Figure 8 shows a ROC graph of true positive rate ($1 - p_{\text{fn}}$) versus false positive rate (p_{fp}) obtained as a result of adjusting C_T for reckless, random and opportunistic attackers given different p_{err}. As we increase C_T, the true positive rate increases while the false positive rate increases. We see that with our specification-based IDS technique, the true positive rate can approach 100% for detecting attackers when using a sufficiently high C_T, i.e., an attacker is always detected with probability 1 without false negatives, while bounding the false positive rate to below 0.05% for reckless attackers, below 7% for random attackers with attack probability as low as 0.2, and and below 6.1% for opportunistic attackers, when $p_{\text{err}} = 1\%$ (the first entry in Table VI). Note the ROC surfaces for random and opportunistic attackers cross over roughly along a curve at $p_{\text{err}} = 0.015$, indicating that when $p_{\text{err}} < 0.015$ an opportunistic attacker with $\varepsilon = 0.8$ is more difficult to be detected than a random attacker with $p_a = 0.2$, and vice versa after $p_{\text{err}} > 0.015$. The highlighted dots show the points on the corresponding surface that meet the maximum p_{fn} requirement (1%) while minimizing p_{fp}. Our adaptive IDS design allows the system to adaptively adjust C_T dynamically to satisfy the imposed p_{fn} requirement while minimizing p_{fp} in response to dynamically changing environment conditions (through p_{err}) and the suspected attacker type detected at runtime (e.g., a random attacker).

V. COMPARATIVE ANALYSIS

In this section, we compare our IDS design with a multitrust anomaly-based IDS called Multi-agent System (MAS) developed by Tsang and Kwong [26] intended for industrial CPSs. MAS includes an analysis function called Ant Colony Clustering Model (ACCM) to reduce the characteristically high false positive rate associated with anomaly-based approaches while minimizing the training period by using an unsupervised approach to machine learning. MAS uses a standard data set, KDD Cup 1999, for testing. We use it as a benchmark against which our IDS is compared for three reasons: First, there is no existing UAV IDS available for performance comparison; industrial process control environments of MAS/ACCM are close to UAV environments with similar safety requirements. Second, MAS/ACCM reported false positive rate and false negative rate data for ease of comparison. Third, unlike anomaly-based IDS approaches, ACCM generated good false positive rates (reported 1 to 6%). We visualize Tsang and Kwong's results in Figure 9.

Figure 10 visually compares the ROC graphs (true positive rate or $1 - p_{\text{fn}}$ versus p_{fp}) for BRUIDS and ACCM. We set $p_{\text{err}} = 0.010$ for UAVs. This is because 1% of mis-monitoring due to ambient noise, temporary system faults and wireless communication faults in airborne environments is reasonable. This is based on Ho and Shimamoto reporting a $0.2 - 7.5\%$ packet error rate (depending on altitude, network size and channel access technique) [11] and Palazzi et al. experimenting with packet error rates between 0.1 and 1.0% [18]. Figure 10 shows for reckless and highly aggressive random attackers,

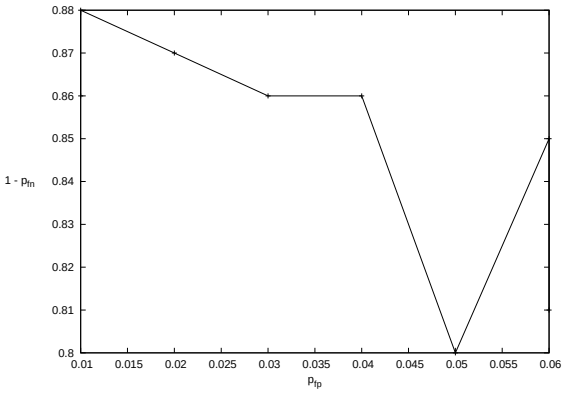

Fig. 9. ACCM Receiver Operating Characteristic Graph.

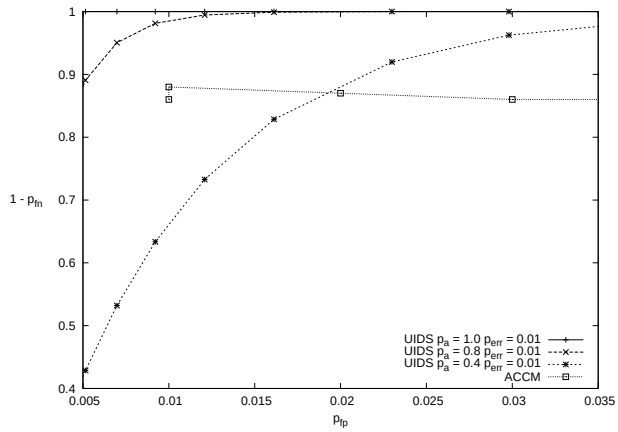

Fig. 11. Detail for AUC Inspection.

BRUIDS outperforms ACCM across the domain of p_{fp}. Also, it shows that BRUIDS outperforms ACCM for $p_{\mathrm{fp}} > 0.02$ for cautious random attackers.

We first note that the coverage area (out of a one by one area) below the ROC curve, referred to as Area Under the Curve (AUC), measures the IDS accuracy. An area of 1 represents a perfect test; an area of 0.5 represents a worthless test.

We clearly see that for BRUIDS with $p_a = 1$ or 0.8, the true positive rate is always higher than ACCM, given the same false positive rate. Therefore the AUC of BRUIDS is clearly greater than that of ACCM for these configurations. For BRUIDS with $p_a = 0.4$ (a more cunning attacker), the detection rate is not always higher than ACCM; however, a closer look through Figure 11 reveals that the AUC of BRUIDS is still greater. Moreover, in the experiment conducted by Tsang and Kwong [26], presumably only reckless attackers were considered in which case BRUIDS with a AUC nearly equal to 1 clearly outperforms ACCM.

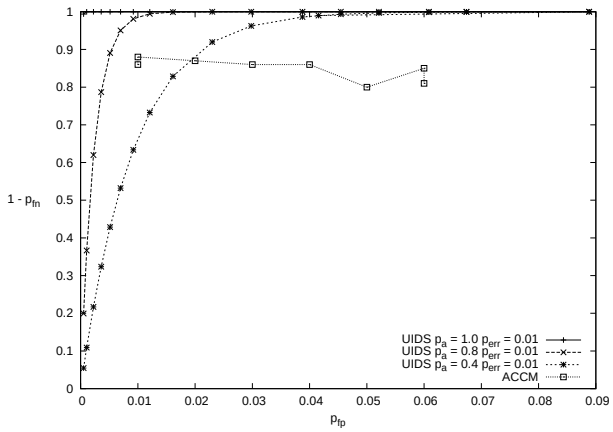

Fig. 10. BRUIDS and ACCM ROC Graph Comparison.

VI. Related Work

The topic of IDS and airborne networks is not widely discussed in the academic literature, so we highlight some related examples that motivated our work. We first survey existing work in airborne systems IDS in the literature. Because airborne system applications may be deployed in mobile ad hoc networks (MANETs) with embedded CPS physical components, we also survey related IDS techniques in MANETs and CPSs.

A. Unmanned Aircraft Systems

Blasch et al. [4] proposes a warplanning situational awareness tool that classifies outsiders in the physical domain as friendly, neutral or belligerent. Specifically, it is not an IDS that identifies insider attackers in the cyber domain. The authors use several metrics to evaluate their classifier: precision, recall, accuracy, activities of interest, timeliness and throughput. They use ROC plots to visualize their classifier's effectiveness: they transform their recall metric into true positive rate and their precision metric into the false positive rate. In particular, Blasch et al. include a 3D ROC plot where classification latency is the third dimension.

Trafton and Pizzi [25] did an investigation which motivated and broadly described the role of intrusion detection in this application but did not propose let alone measure a solution. The authors proposed Joint Airborne Network Services Suite (JANSS) which provides a framework to integrate an airborne military network. Among other services, JANSS covers intrusion detection. They describe IDS as part of a larger Information Assurance strategy. They only say that the IDS should be host-based; the alternative is a network-based IDS.

Lauf and Robinson [14] investigated Distributed Apt Resource Transference System (DARTS). DARTS is an intrusion tolerance strategy that reallocates resources to tolerate faults and attacks. It built on their prior work, HybrIDS [15], which prompts (triggers) resource reallocation. The key innovation in DARTS is its service discovery protocol (SDP): It combines an on-demand flooding approach with a gossip approach to get the benefits while masking the drawbacks of each. The drawback of an on-demand flooding SDP is an intractable burst of communications. The drawback of a gossiping SDP is staleness. The authors measure the effectiveness of DARTS using communications overhead (messages exchanged for reallocation) and downtime (time to reallocation).

HybrIDS is an anomaly-based approach. Specifically, HybrIDS comprises two semi-supervised approaches resulting in three operational phases: MDS training, MDS testing/CCDS training and CCDS testing. They chose a behavior-based approach rather than a traffic-based approach due to time and memory constraints of an embedded system. HybrIDS is distributed for scalability. HybrIDS comprises two intrusion detection methods: Maxima Detection System (MDS) and Cross-Correlative Detection System (CCDS). MDS detects single intruders after a short training phase and accomplishes a more in-depth training phase for CCDS. CCDS can detect cooperating intruders after the longer training phase provided by MDS. Lauf et al. measure the performance of HybrIDS using pervasion, which they define as the percentage of malicious nodes in the system. The authors could detect intruders even with a 22% pervasion; a Byzantine fault model establishes a theoretic limit of 33%. During the training/MDS phase, they collect data regarding system state. They sequence the nominal system states for use by CCDS so that the probability density function (PDF) resembles a chi-squared distribution. Lauf et al. [15] use applications' system call history as their audit data. The authors identify two parameters to create an effective IDS for a resource constrained application: audit collection period [data collection cycle (DCC)] and audit analysis period [data processing cycle (DPC)]. A longer DCC increases the memory stress while increasing the detection accuracy of an intrusion detector, and a shorter DPC increases the processor stress while decreasing the detection latency of an intrusion detector. No analysis was given regarding the tradeoff between DCC and DPC. More importantly, [14], [15], [25] did not report false negative rate p_{fn} (i.e., missing a malicious node) and the false positive rate p_{fp} (i.e., misidentifying a normal node as a malicious node).

While we address which type of style the player (UAV-target jamming) is using, in [22], Shen et al. use game theory to model the different types of approaches. The authors use their model to generate flight plans for a fleet of UAVs that will maximize target coverage and UAV survivability. Their model has three levels: object, situation and threat. Shen et al. consider cooperative effects of defending nodes by distinguishing self-protection from support jamming. In [21], Shen et al. consider six types of attack: buffer overflow, semantic URL attack, e-mail bombing, e-mail spam, malware attachment and DoS. The authors consider four defensive measures: IDS deployment, firewall configuration, email-filter configuration and server shutdown or reset.

B. Ad Hoc Networks

Existing IDS techniques for MANETs are centered around secure routing, using monitoring techniques to detect deviation of normal behaviors in data routing or forwarding. Often, specific MANET routing protocols such as Dynamic Source Routing (DSR) and Ad hoc On-Demand Distance Vector (AODV) are being considered in IDS design.

Bella et al. [3] propose a reputation-based IDS that bases node reputation on the energy it uses for others in comparison with the energy it uses for itself: specifically, the ratio of packets forwarded to packets sourced. They calculate aggregate reputation score as the weighted sum of the locally observed reputation score, the Neighbor Reputation Table (NRT) value, historical global reputation score, the Global Reputation Table (GRT) value and a third party recommendation; the design ages scores such that the reputation of inactive nodes deteriorates. One con of this study is that nodes that do not have a demand for forwarding will be penalized unfairly. Moreover, only reputation scores over time for normal, selfish and malicious nodes were reported, without providing false positive rates, true positive rates or accuracy data. Their design is geared toward detection of misbehaving nodes, rather than detection of compromised nodes.

Buchegger and Le Boudec [5] propose a distributed IDS called CONFIDANT which extends dynamic source routing (DSR) by measuring reputation with "no forwarding" behavior. The authors distinguish three levels of multitrust: *experienced* data is a firsthand account which has the most weight, *observed* data which has less weight than experienced data happens in the neighborhood (within radio range) and *reported* data which has less weight than experienced or observed data is an account coming from outside the neighborhood. Borrowing from the field of ecology, they classify nodes into one of three categories: suckers (who always assist neighbors), cheats (who never assist neighbors) and grudgers (who assist neighbors until they experience non-reciprocation). One strength of this study is the capability for reformed or falsely detected nodes to rejoin the network. Buchegger and Le Boudec measured packet drops, packet drop rate, goodput (which they define as packets received over packets sourced), throughput and overhead; time, network size and level of network hostility were their independent variables. Again, no data were reported on false positive rates, true positive rates or accuracy data in this study.

Michiardi and Molva [16] propose an IDS called Collaborative Reputation Mechanism (CORE). Neighbors of a suspect calculate its *subjective* reputation score from experience of some property f (for example, DSR routing or packet forwarding) weighting earlier and later observations differently, and nodes calculate a suspect's *functional* reputation over multiple f weighting various f differently and aging (decreasing over time) the reputations of inactive nodes. In CORE, each node regards every other node as either trusted (positive reputation) or misbehaving (negative reputation); nodes deny service requests and ignore reputation information from misbehaving nodes. Strengths of this study are the toleration of slander attacks and distinct sanctions for selfish and malicious nodes. Michiardi and Molva did not report any numerical data.

Tseng et al. [27] use an AODV-based finite state machine (FSM) to establish a specification for a traffic-based IDS. Distributed network monitors maintain an FSM for each routing transaction (request and reply). States are normal, alarm or suspicious; in suspicious states, the network monitor asks its peers for additional audit data for the transaction. The work is specific to AODV for secure routing. However, no data were reported on false positive and negative probabilities.

Hadjichristofi et al. [10] studied an integrated management framework for MANET which encompasses routing, security, resource management and network monitoring functions. The

authors propose a novel routing protocol that informs the resource management function. The security function includes a trust management approach. The trust management approach begins with authentication and updates scores based on the trusted node's behavior.

Kiess and Mauve [12] survey real-world implementations of MANETs in order to identify viable test beds for MANET research. The authors motivate this study by highlighting the shortcomings of MANET simulations and emulations.

C. CPSs

Another broader research area that could encompass intrusion detection for airborne networks and communications are CPSs. CPSs typically have multiple control loops, strict timing requirements, a wireless network segment, predictable network traffic and contain physical components [23]. CPSs fuse cyber (network components and commodity servers) and physical (sensors and actuators) domains. They may contain human actors and mobile nodes. The focuses for CPS IDSs are leveraging unique CPS traits (sensor inputs, algorithms and control outputs) and detecting unknown attacks.

Porras and Neumann [19] study a hierarchical multitrust behavior-based IDS called Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD) using complementary signature based and anomaly-based analysis. The authors identify a signature-based analysis trade between the state space created/runtime burden imposed by rich rule sets and the increased false negatives that stem from a less expressive rule set. Porras and Neumann highlight two specific anomaly-based techniques using statistical analysis: one studies user sessions (to detect live intruders), and the other studies the runtime behavior of programs (to detect malicious code). EMERALD provides a generic analysis framework that is flexible enough to allow anomaly detectors to run with different scopes of multitrust data (service, domain or enterprise). However, Porras and Neumann did not report false positive or false negative rate data.

Tsang and Kwong [26] propose a multitrust IDS called Multi-agent System (MAS) that includes an analysis function called Ant Colony Clustering Model (ACCM). The authors intend for ACCM to reduce the characteristically high false positive rate of anomaly-based approaches while minimizing the training period by using an unsupervised approach to machine learning. MAS is hierarchical and contains a large number of roles: monitor agents collect audit data, decision agents perform analysis, action agents effect responses, coordination agents manage multitrust communication, user interface agents interact with human operators and registration agents manage agent appearance and disappearance. Their results indicate ACCM slightly outperforms the true positive rates and significantly outperforms the false positive rates of k-means and expectation-maximization approaches. Because of the good false positive results reported, i.e., the ACCM false positive rate ranges from 1 to 6%, we use [26] as a benchmark in our comparative analysis in Section V.

Ying et al. [30] model fault diagnosis related to a medical CPS with a Hidden Markov Model. There are critical differences between fault diagnosis and intrusion detection. Faults may be uniformly or normally distributed, while cyber attacks are not random. Faults do not seek to evade detection, while intruders do. While faults cannot be attributed to a human actor, intrusions can.

VII. Lessons Learned

One practical consideration is that behavior rules are directly derived from threats. Hence, the threat model must be broad enough to cover all possible threats that exploit system vulnerabilities. This places the responsibility for developing a complete attack model with the system designers. When a threat is overlooked, the state machine will lack unsafe states associated with the overlooked attack behavior indicator, and the attack will go undetected by BRUIDS. A second consideration is that when new threats are discovered and introduced to the threat model, new behavior rules corresponding to the new threats must be added to the rule set because behavior rules are derived directly from threats. BRUIDS allows newly identified threats to be added to the threat model and hence the corresponding new behavior rules to be derived from which the state machine is automatically generated for intrusion detection. Finally, while BRUIDS can adaptively adjust the detection strength in terms of the C_T value to satisfy the maximum p_{fn} requirement while providing a p_{fp} as small as possible given knowledge of the attacker type and p_{err}, determination of the attacker type and p_{err} with precision at runtime deserves more research efforts.

VIII. Conclusions

For UAVs, being able to detect attackers while limiting the false positive rate is of utmost importance to protect the continuity of operation. In this paper we proposed an adaptive behavior-rule specification-based IDS technique for intrusion detection of compromised UAVs using an applied rule set derived from the UAV threat model. We demonstrated that the true positive rate approaches one (that is, we can always catch the attacker without false negatives) while bounding the false positive rate to below 0.05% for reckless attackers, below 7% for random attackers with attack probability as low as 0.2 and below 6% for opportunistic attackers, when the error of monitoring due to environment noise is at 1%. Through a comparative analysis, we demonstrated our behavior-rule specification-based IDS technique outperforms an existing multitrust anomaly-based IDS approach in detection accuracy.

In the course of this study, we identified a number of future research areas. The first is to consider additional performance metrics such as detection latency. The second open area concerns the attack model. One can consider additional insider attack behaviors such as insidious attackers that lie in wait until they have co-opted enough nodes to launch a devastating attack. The third line of investigation concerns the defender's response [17]. For example, the IDS can perform better if it retunes its parameters (such as compliance threshold, audit interval and state machine granularity) based on the type and strength of adversary it faces.

REFERENCES

[1] M. Aldebert, M. Ivaldi, and C. Roucolle. Telecommunications Demand and Pricing Structure: An Econometric Analysis. *Telecommunication Systems*, 25:89–115, 2004. 10.1023/B:TELS.0000011198.50511.a4.

[2] F. B. Bastani, I. R. Chen, and T. W. Tsao. Reliability of systems with fuzzy-failure criterion. In *Annual Reliability and Maintainability Symposium*, pages 442–448, Anaheim, California, USA, January 1994.

[3] G. Bella, G. Costantino, and S. Riccobene. Managing Reputation over MANETs. In *Fourth International Conference on Information Assurance and Security*, pages 255–260, Naples, Italy, September 2008.

[4] E. Blasch, J. Salerno, and G. Tadda. Measuring the worthiness of situation assessment. In *The National Aerospace and Electronics Conference*, volume Dayton, OH, USA, pages 87–94, July 2011.

[5] S. Buchegger and J.-Y. Le Boudec. Performance analysis of the CONFIDANT protocol. In *3rd international symposium on Mobile ad hoc networking & computing*, pages 226–236, Lausanne, Switzerland, June 2002.

[6] I. R. Chen and F. B. Bastani. Effect of artificial-intelligence planning-procedures on system reliability. *IEEE Transactions on Reliability*, 40(3):364–369, 1991.

[7] I. R. Chen, F. B. Bastani, and T. W. Tsao. On the reliability of AI planning software in real-time applications. *IEEE Transactions on Knowledge and Data Engineering*, 7(1):4–13, 1995.

[8] I. R. Chen and T. H. Hsi. Performance analysis of admission control algorithms based on reward optimization for real-time multimedia servers. *Performance Evaluation*, 33(2):89–112, 1998.

[9] S. Gorman, Y. J. Dreazen, and A. Cole. Insurgents Hack U.S. Drones. *Wall Street Journal*, page A.1, December 2009.

[10] G. C. Hadjichristofi, L. A. DaSilva, S. F. Midkiff, U. Lee, and W. D. Sousa. Routing, security, resource management, and monitoring in ad hoc networks: Implementation and integration. *Computer Networks*, 55(1):282 – 299, 2011.

[11] D.-T. Ho and S. Shimamoto. Highly reliable communication protocol for WSN-UAV system employing TDMA and PFS scheme. In *Global Communications Conference Workshops*, pages 1320–1324, Houston, TX, USA, December 2011.

[12] W. Kiess and M. Mauve. A survey on real-world implementations of mobile ad-hoc networks. *Ad Hoc Networks*, 5(3):324 – 339, 2007.

[13] A. Kim, B. Wampler, J. Goppert, I. Hwang, and H. Aldridge. Cyber Attack Vulnerabilities Analysis for Unmanned Aerial Vehicles. In *Infotech@Aerospace*, Garden Grove, CA, USA, June 2012.

[14] A. Lauf and W. Robinson. Fault-tolerant distributed reconnaissance. In *Military Communications Conference*, pages 1812–1817, San Jose, CA, USA, November 2010.

[15] A. P. Lauf, R. A. Peters, and W. H. Robinson. A distributed intrusion detection system for resource-constrained devices in ad-hoc networks. *Ad Hoc Networks*, 8(3):253–266, 2010.

[16] P. Michiardi and R. Molva. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *International Federation for Information Processing TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, pages 107–121, Portoroz, Slovenia, September 2002.

[17] R. Mitchell and I. R. Chen. Effect of Intrusion Detection and Response on Reliability of Cyber Physical Systems. *IEEE Transactions on Reliability*, 62(1):199–210, March 2013.

[18] C. E. Palazzi, C. Roseti, M. Luglio, M. Gerla, M. Y. Sanadidi, and J. Stepanek. Enhancing Transport Layer Capability in HAPS-Satellite Integrated Architecture. *Wireless Personal Communications*, 32:339–356, 2005. 10.1007/s11277-005-0751-2.

[19] P. Porras and P. Neumann. EMERALD: Event monitoring enabling responses to anomalous live disturbances. In *20th National Information Systems Security Conference*, pages 353–365, Baltimore, MD, USA, October 1997.

[20] S. M. Ross. *Introduction to Probability Models, 10th Edition*. Academic Press, 2009.

[21] D. Shen, G. Chen, E. Blasch, and G. Tadda. Adaptive Markov Game Theoretic Data Fusion Approach for Cyber Network Defense. In *Military Communications Conference*, pages 1–7, Orlando, FL, USA, October 2007.

[22] D. Shen, G. Chen, J. Cruz, and E. Blasch. A Game Theoretic Data Fusion Aided Path Planning Approach for Cooperative UAV ISR. In *Aerospace Conference*, pages 1–9, Big Sky, MT, USA, March 2008.

[23] S. Shin, T. Kwon, G.-Y. Jo, Y. Park, and H. Rhy. An Experimental Study of Hierarchical Intrusion Detection for Wireless Industrial Sensor Networks. *IEEE Transactions on Industrial Informatics*, 6(4):744–757, November 2010.

[24] X. Tian, Y. Bar-Shalom, G. Chen, E. Blasch, and K. Pham. A unified cooperative control architecture for UAV missions. In *Defense, Security, and Sensing*, pages 83920X–83920X, Baltimore, MD, USA, April 2012. International Society for Optics and Photonics.

[25] R. Trafton and S. Pizzi. The Joint Airborne Network Services Suite. In *Military Communications Conference*, pages 1–5, Washington, DC, USA, October 2006.

[26] C.-H. Tsang and S. Kwong. Multi-agent intrusion detection system in industrial network using ant colony clustering approach and unsupervised feature extraction. In *International Conference on Industrial Technology*, pages 51–56, Hong Kong, December 2005.

[27] C.-Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt. A specification-based intrusion detection system for AODV. In *1st workshop on Security of ad hoc and sensor networks*, pages 125–134, Fairfax, VA, USA, October 2003.

[28] M. Yampolskiy, P. Horvath, X. Koutsoukos, Y. Xue, and J. Sztipanovits. Systematic analysis of cyber-attacks on CPS-evaluating applicability of DFD-based approach. In *5th International Symposium on Resilient Control Systems*, pages 55–62, Salt Lake City, UT, USA, August 2012.

[29] O. Yilmaz and I. R. Chen. Utilizing call admission control for pricing optimization of multiple service classes in wireless cellular networks. *Computer Communications*, 32(2):317 – 323, 2009.

[30] J. Ying, T. Kirubarajan, K. Pattipati, and S. Deb. A hidden Markov model based algorithm for online fault diagnosis with partial and imperfect tests. In *Systems Readiness Technology Conference*, pages 355–366, San Antonio, TX, USA, August 1999.