Reliability of Autonomous IoT Systems with Intrusion Detection Attack-Defense Game Design

Ding-Chau Wang[†], Ing-Ray Chen[‡], and Hamid Al-Hamadi*

Abstract — **In this paper we develop an intrusion detection attack-defense game for IoT systems for which autonomous IoT devices collaboratively solve a problem. We develop an analytical model to determine the conditions under which malicious nodes have no incentives to perform attack in the intrusion detection attack-defense game. We also develop a stochastic Petri net model to analyze the effect of attack-defense behaviors on system reliability, given a definition of system failure conditions as input. The performance evaluation results demonstrate that our IDS attack-defense game design greatly improves system reliability over existing autonomous IoT systems without gaming design consideration when attacks are reckless and intensive.**

Keywords— Intrusion detection; attack-defense games; Internet of Things; autonomous systems; reliability.

I. Introduction

With the proliferation of Internet of Things (IoT) devices, we have witnessed the era of autonomous IoT-based applications, including parking space finding [1], participatory sensing of air quality [2], smart service community [3, 4], crowdsensing for cooperative problem solving [5, 6], smart Internet of vehicles (IoV) information systems [7], IoT-embedded cyber physical systems (CPS) [8, 9, 10, 11], etc. All these applications involve autonomous IoT devices collaborating with each other for problem solving or decision making.

The most important requirement of such autonomous IoT systems is that information supplied from collaborating IoT devices must be trustworthy based on which data analysis may be performed to solve a problem or make a correct decision. Consequently, a central issue is whether certain IoT devices are malicious in supplying false information for own benefits or whether a group of malicious nodes collude with each other for group benefits. Since potentially there will be a huge number of IoT devices, it is highly impractical to use a centralized entity (say sitting in the cloud) to perform intrusion detection to filter out untrustworthy information, since the centralized entity cannot physically perform misbehavior detection itself and needs to collect misbehavior reports/logs from IoT devices. This will not only introduce a large amount of traffic between IoT devices and the centralized entity thus crippling the IoT communication network, but also consume energy of resource constrained IoT devices. Hence, distributed misbehavior detection is the only feasible way for autonomous IoT systems, with the centralized entity performing auditing when necessary.

In this paper we develop a lightweight intrusion detection system (IDS) attack-defense game for detecting malicious IoT devices in autonomous IoT systems. The basic idea of our lightweight IDS game design is that the system does auditing only occasionally controlled by an auditing probability, while leaving intrusion detection to IoT devices themselves in a distributed manner. The game's outcome is the system reliability measured by the system's mean time to failure (MTTF), when given a definition of system failure conditions as input.

We design our IDS attack-defense game following the design principle of mechanism design theory (also called reverse game theory) [12] such that every node in the system must participate in game playing so that nodes are provided with incentives and act in such a way to further the interest of the designer, despite the fact that nodes are strategic and self-interested, and possess private information [13]. Designing a game to motivate users to follow the prescribed rules has been widely applied to communication system design including cognitive radio networks [14] and vehicular networks [5]. To the best of our knowledge, this is the first work for IoT IDS design.

The basic idea for our lightweight distributed IDS game is that a target node is periodically being voted on by a group of neighbor nodes to determine if the target node is good or bad (i.e., malicious). The defense system (presumably sitting in the cloud) can optionally audit the voting outcome to detect if IDS voting is performed faithfully and correctly. A node, if invited to determine if a target node is good or bad, uses its basic host-level IDS functions characterized by a host-level false positive rate and a host-level false negative rate, to cast a "yes" (meaning the target node is good) or "no" (meaning the target node is bad) vote. The outcome of IDS voting, gathered by the group of voting members, shall determine if the target node should be evicted or retained in the system, and is reported to the defense system. Since there may be several attackers (i.e., insiders) during an IDS voting process, they can collude with each other such that if the target node is a good node, they vote "no" against the good target node so as to evict the good target node from the system, and, conversely, when the target node is a bad node, they vote "yes" for the bad target node so as to keep the bad target node from being evicted. To punish

[†] *Ding-Chau Wang is with Department of Information Management, Southern Taiwan University of Science and Technology, Tainan, Taiwan; Email: dcwang@stust.edu.tw*
[‡] *Ing-Ray Chen is with the Department of Computer Science, Virginia Tech, Falls Church, VA 22043, USA; Email: irchen@vt.edu*
**Hamid Al-Hamadi is with the Department of Computer Science, Kuwait University, Kuwait; Email: hamid@cs.ku.edu.kw*

such misbehavior, the defense system can perform auditing (controlled by an auditing probability parameter) after each IDS voting event to obtain the true outcome and penalize nodes who cast a different vote from the auditing outcome. This forces every malicious node to decide whether it should attack or not attack in an IDS voting cycle, especially if the penalty is severe at the designer's choice. The analytical model developed in this paper shall allow a designer to determine the best penalty and the best auditing probability for maximizing the system reliability based on the performance and reliability characteristics of the autonomous IoT system in hand.

Our analytical model aims to determine the condition under which malicious nodes have no incentives to perform attack during IDS voting in our intrusion detection attack-defense game. The condition, characterized by a set of loss and gain payoff functions as well as the attacker's attack probability and the defender's auditing probability, is analytically derived in the paper. We illustrate how our IDS attack-defense game can be applied to an autonomous mobile cyber physical system (CPS) wherein each node is given a "life quota" by parameterizing (i.e., giving value to) the loss and payoff functions such that there exists a minimum auditing probability after which an attacker would be discouraged to attack during IDS voting so as to maximize its own payoff. We further develop a performance model to analyze the effect of attack-defense behaviors as well as the attacker's attack probability and the defender's auditing probability on system reliability.

Our work has the following unique contributions:

1. We develop IDS attack-defense games that must be played by every node of an autonomous IoT system. We derive the exact condition under which malicious nodes have no incentives to perform attack in the IDS attack-defense game as well as the best defender setting to maximize the system reliability, when given a definition of system failure conditions as input.
2. We develop an analytical model based on Stochastic Petri Net (SPN) modeling techniques (e.g., [15-19]) to describe the IDS attack-defense game dynamics. The SPN model allows one to analyze the effect the attack probability (by an attacker), the auditing probability (by the defense system), and the penalty (to apply to nodes whose vote mismatches with the auditing outcome) on system reliability.
3. We put our IDS attack-defense game into practical use by applying it to an autonomous mobile CPS [11] wherein each node is given a "life quota" for it to remain in the system. We compare the performance of our IDS game with that of baseline the mobile CPS [11] without game design.

The rest of the paper is organized as follows: Section II discusses the system model for IDS voting game playing. Section III describes in detail of our IDS voting game design and analytically derives the condition under which malicious nodes have no incentives to perform attack as well as the best defender setting to prolong the system lifetime. Section IV applies our IDS attack-defense game to a baseline CPS application wherein each node is given a "life quota" for it to remain in the system and develops an SPN model based on SPNP [15] to analyze the effect of attack-defense strategies played by attackers/defenders on system reliability. Section V provides numerical results including a comparative analysis of our IDS game against the baseline IoT system without game design. Finally, Section VI summarizes the paper and outlines future research areas.

II. System Model

We first discuss system failure conditions for an autonomous IoT system based on which the system MTTF is derived. Then, we discuss the system model for the attack-defense IDS game.

A. System Failure Conditions

The following failure conditions can cause an autonomous IoT system to fail:

- Byzantine failure [20] occurs when one-third or more of the nodes are compromised. The reason is that once an autonomous system contains at least 1/3 compromised nodes, it is impossible to reach a consensus, hence inducing a system failure.
- Energy depletion failure occurs when energy is too depleted to be able to accomplish the mission. This is especially critical for an autonomous collaborative IoT system that must complete the mission within a deadline without energy replenishment.

B. IDS Attack-Defense Game

The attacker behavior comes in two forms. The first form of attacker behavior derives from "capture" attacks to compromise nodes, i.e., to turn a good node into a bad node. This is especially true for sensor/actuator IoT devices that do not have proper physical protection and can be easily physically captured by intruders and converted into malicious nodes (i.e., inside attackers). It is possible that viruses can also invade good nodes and turn them into malicious nodes. We assume a per-node capture rate of λ. The second form of attacker behavior derives from insider attacks during IDS voting. An insider may only attack probabilistically to evade detection. That is, a malicious node decides to attack with probability P_a and not to attack with probability $1 - P_a$ during IDS voting. A goal of our IDS game design is to discourage malicious nodes from performing attacks such that $P_a = 0$ at which the system can obtain the maximum lifetime.

Table I: IDS Attack-Defense Game Payoff.

Defender Strategies	Attacker Strategies	
	Attack with probability P_a	Not attack with probability $1-P_a$
Audit with probability P_c	$L_a^c - C, -L_a^c$	$L_{na}^c - C, -L_{na}^c$
Not audit with probability $1-P_c$	$-G_a^{nc}, G_a^{nc}$	$-G_{na}^{nc}, G_{na}^{nc}$

The defense behavior also comes in two forms. The first form of defense is at the host level. At the host IDS level, node i monitors positive and negative experiences it has toward node j when it encounters with node j (for immobile IoT devices node i and node j would be neighbors within detection range) to judge if node j complies with prescribed protocol execution. Anomaly detection techniques including discrepancy of voting results during IDS voting may be used for this purpose. Node i can use Beta (a, b) distribution [9] to model the compliance degree of node j in the range of (0, 1) as a random variable where a and b represent the numbers of positive and negative experiences respectively, such that the estimated mean compliance degree is $a/(a+b)$. If node j's compliance degree is less than a minimum compliance degree C_T, node i considers node j as bad; otherwise node i considers node j as good. The minimum compliance degree C_T therefore decides the host-level false negative probability H_{pfn} and the host-level false positive probability H_{pfp}. We assume that each node is thoroughly tested for its host-level intrusion capability before it is released to operational use. Hence, H_{pfn} and H_{pfp} are provided as input.

The second form of defense behavior is at the system-level via IDS voting for which the detection strength is controlled by the number of voters (m) and how often intrusion detection is performed (in every T_{IDS} interval). In each IDS voting cycle, m nodes which are neighbors of a target node will participate in IDS voting to vote for or against the target node, based on host-level IDS outcomes. If the majority voting outcome is "no" then the target node is evicted; otherwise, the target node is retained. To preserve energy of IoT nodes, the defense system will audit the voting outcome with probability P_c and will not audit with probability$1-P_c$. To punish misbehavior during IDS voting, the defense system penalizes nodes who cast a different vote from the auditing outcome, the severity of which is to be determined by the system designer when the IDS attack-defense game is setup.

III. IDS Attack-Defense Game

In this section, we formulate the IDS attack-defense game based on *mechanism design theory* (also called reverse game theory) [12] to model decision making between the attacker and the defense system and then present a theoretical analysis. The game models the relationship between the defense system and a malicious node i who has two options: attack or not attack during IDS voting. On the other side, from the defense system's perspective, it decides to audit the voting result with probability P_c or not to audit with probability $1-P_c$.

The payoff matrix for the defense system and a malicious node i in the game model is shown in Table I. The table entry is in the format of (defense system payoff, malicious node i payoff). For example, if the defense system checks the voting result while malicious node i dishonestly reports a fake report the payoff to the defense system is $L_a^c - C$ and the payoff to malicious node i is $-L_a^c$.

We explain the payoff matrix Table I below.

According to the described game model, during IDS voting (to determine if a target node is malicious), a malicious node i can attack with probability P_a and not attack with probability $1-P_a$. If it decides to attack, it will cast a "no" vote against a good target node and a "yes" node for a bad target node, with the "no" vote meaning that the target node is a bad node and the "yes" vote meaning that the target node is a good node. If it decides not to attack, it will behave like a good node so it will cast a vote as what a good node would do based on its basic host-level IDS function. On the other hand, the defense system decides to audit the voting outcome with probability P_c and not to audit the voting outcome with probability $1-P_c$. Auditing is an expensive operation. We denote the cost by C. The system will have to collect relevant information from all nodes who have had experiences with the target node. If the target node is relatively immobile, the set of relevant nodes may be small but for a highly mobile node, the system may have to probe all nodes in the system who have had experiences with the target node. The high cost is unavoidable in order to ensure that "the auditing outcome" reflects "the true outcome" of whether the target node is malicious or not.

There are 4 cases, as described in Table I:

- Case 1: The defense system decides to audit the voting outcome with probability P_c and a malicious node i decides to attack with probability P_a. The defense system can detect the true outcome (that is if the target node is good or bad) by collecting reports from all relevant nodes in the system and then punish the nodes who cast a different vote with a penalty denoted by $L_a^c \geq 0$ with the superscript "c" meaning that the system "checks" the voting outcome, and the subscript "a" meaning that the malicious node "attacks" during IDS voting. The loss to the malicious node is treated as a gain to the defense system. Therefore, the payoffs to the defense system and malicious node i are $L_a^c - C$ and $-L_a^c$ respectively.
- Case 2: The defense system decides to audit the voting outcome with probability P_c and a malicious node i decides not to attack with probability $1-P_a$. Again

the defense system can detect the true outcome (that is if the target node is good or bad) and then punish the nodes who cast a different vote. Since malicious node i acts as if it is a good node and it casts the right vote as a normal node would do, it would not receive a penalty. Because the defense system performs a thorough audit of the voting outcome and follows the true voting outcome, the defense system gains something positive in reliability denoted by $L_{na}^{c} \geq 0$ with the superscript "c" meaning that the system "checks" the voting outcome, and the subscript "na" meaning that the malicious node does "not attack" during IDS voting. In practice the gain may be small. Nevertheless, the gain to the defense system is a loss to the malicious node. Therefore, the payoffs to the defense system and malicious node i are $L_{na}^{c} - C$ and $-L_{na}^{c}$ respectively.

- Case 3: The defense system decides not to audit the voting outcome with probability $1 - P_c$ and a malicious node i decides to attack with probability P_a. Since the defense system does not audit the voting outcome, the voting outcome will be accepted as is which may impact the system reliability. Let the impact be represented by $G_{a}^{nc} \geq 0$ (with the superscript "nc" meaning that the system does "not check" the voting outcome, and the subscript "a" meaning that the malicious node "attacks" during IDS voting) which can be considered as a gain to malicious node i. The gain to malicious node i is treated as a loss to the defense system. Therefore, the payoffs to the defense system and malicious node i are $-G_{a}^{nc}$ and G_{a}^{nc} respectively.
- Case 4: The defense system decides not to audit the voting outcome with probability $1 - P_c$ and a malicious node i also decides not to attack with probability $1 - P_a$. In this case the defense system again does not audit the voting outcome, the voting outcome will be accepted as is which may adversely impact the system reliability. Let the impact be represented by $G_{na}^{nc} \geq 0$ (with the superscript "nc" meaning that the system does "not check" the voting outcome, and the subscript "na" meaning that the malicious node does "not attack" during IDS voting) which can be considered as a gain to malicious node i. The gain to malicious node i is treated as a loss to the defense system. Therefore, the payoffs to the defense system and malicious node i are $-G_{na}^{nc}$ and G_{na}^{nc} respectively.

Theorem 1: To discourage malicious node i from performing attacks during IDS voting, the following condition must satisfy: $P_c(L_{a}^{c} - L_{na}^{c}) \geq (1 - P_c)(G_{a}^{nc} - G_{na}^{nc})$.

Proof: According to our game model and the payoff matrix shown in Table I, if a malicious node does not perform attacks during IDS voting, then its payoff is given by:

$$PAYOFF_{na} = -P_c L_{na}^{c} + (1 - P_c)\, G_{na}^{nc} \tag{1}$$

On the other hand, if a malicious node performs attacks, then its payoff is given by:

$$PAYOFF_{a} = -P_c L_{a}^{c} + (1 - P_c)\, G_{a}^{nc} \tag{2}$$

To guarantee a malicious node i does not have the incentives to perform attacks during IDS voting, we have $PAYOFF_{na} \geq PAYOFF_{a}$, i.e.,

$$-P_c L_{na}^{c} + (1 - P_c) G_{na}^{nc} \geq -P_c L_{a}^{c} + (1 - P_c)\, G_{a}^{nc} \tag{3}$$

Or

$$P_c(L_{a}^{c} - L_{na}^{c}) \geq (1 - P_c)(G_{a}^{nc} - G_{na}^{nc}) \tag{4}$$

■

Theorem 1 above provides a general rule for the design of the loss and gain payoff functions such that a malicious node will not have incentives to perform attacks in our game setting. By rearranging Equation (4), we have:

$$P_c \geq \frac{(G_{a}^{nc} - G_{na}^{nc})}{(G_{a}^{nc} - G_{na}^{nc}) + (L_{a}^{c} - L_{na}^{c})} \tag{5}$$

Condition (5) dictates that the defense system auditing probability P_c must be at least greater than the outcome of the right-hand side expression to discourage malicious nodes from performing attacks during IDS voting. The right-hand side expression outcome depends on the L and G payoff functions which can be publicize to all nodes such that a malicious node will have no incentive of performing attacks during IDS voting. In this paper we investigate a simple "life quota" system to parameterize the L and G payoff functions and the minimum system auditing probability for satisfying Condition (5).

IV. Modeling and Analysis

We apply our IDS attack-defense game to an autonomous IoT system wherein each node is given a "life quota" for it to retain as a member in the system. We also develop an SPN model to analyze the effect of attack-defense strategies played by attackers/defenders on system reliability.

A. Life Payoffs in the IDS Attack-Defense Game

Our life quota system initially allocates a life quota of 1 to every node. When the life quota is reduced to zero because of penalties being applied by the defense system, a node is identified as malicious and is removed from the system. The speed at which a node's life quota is reduced is driven by a life quota decay parameter β which is the fraction of life quota taken away from a node should a penalty is being assessed by the defense system. For example, if β is $1/2$ then a node's life quota is reduced to zero after 2 penalties are being applied to the node. This parameter allows the system designer to adjust the severity of penalty depending on the system requirement. For the most secure system, β can be set to 1, so a single penalty will evict a malicious node.

Based on our proposed life quota scheme, we can parameterize (i.e., give values to) the L and G payoff functions as follows:

- G_a^{nc}: This payoff is applied to a malicious node that performs attack (so it votes no against a good node or yes for a bad node during IDS voting) without being detected because the system does not perform auditing. This payoff is set to 1 because the highest impact achievable by a malicious node is that a good node is voted down and is therefore evicted (so a loss of life quota of 1), or a bad node is voted up and is therefore kept in the system (so a gain of life quota of 1).
- L_a^c: This payoff is applied when a malicious node is detected as having cast a vote that is different from the auditing outcome. It is set to be equivalent to the life quota decay parameter β.
- L_{na}^c: The payoff is applied when a malicious node decides not to attack while the system decides to perform auditing. This payoff is set to zero because a malicious node who decides not to attack will not be penalized with a reduction of life quota.
- G_{na}^{nc}: This payoff is applied to a malicious node that decides not to attack (so it acts like a good node to vote yes for a good node, or no against a bad node during IDS voting) while the system decides not to perform auditing of the IDS voting outcome. This payoff is set to zero as well because the malicious nodes does not gain anything as it does not contribute to misdetection, which happens due to intrinsic imperfect host-level false negative probability H_{pfn} and host-level false positive probability H_{pfp}.

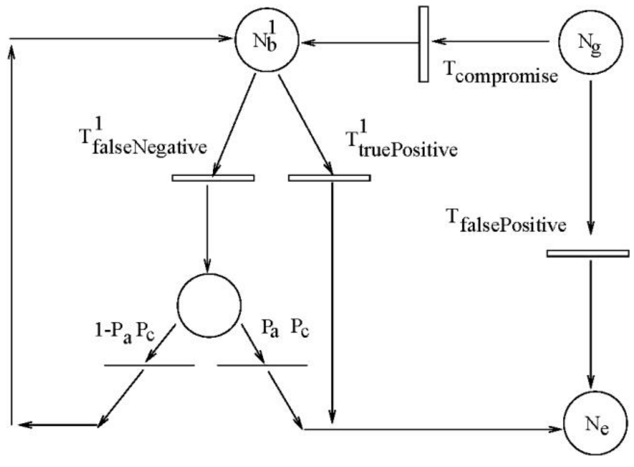

Figure 1: SPN Model for β=1.

With the L and G payoff functions defined as above, from Condition (5), we can set the minimum system auditing probability (denoted byP_c^{min}) as $1/(1+\beta)$ for satisfying Condition (5).

B. Analyzing the Attack-Defense Game Design

To analyze the effect of attack-defense strategies played by attackers/defenders on system reliability, we develop an analytical model based on SPNP [15] to capture IDS game dynamics. Figure 1 shows the SPN model for the case in which $\beta = 1$ such that a single mismatch of the vote cast by a node during IDS voting against the auditing vote outcome will drain the life quota of the node and evict it from the system.

The SPN model is constructed as follows:

- We use places to hold tokens each representing a node. Initially, all N nodes are good nodes and put in place N_g as tokens.
- Good nodes may become compromised with per-node compromising rate λ. This is modeled by associating transition $T_{compromise}$ with an aggregate rate $N_g \times \lambda$. Firing $T_{compromise}$ will move tokens one at a time (if it exists) from place N_g to place N_b. Tokens in place N_b represent compromised but undetected nodes. The superscript of "1" on N_b means that it holds bad nodes with a life quota of 1.
- Good nodes can be misidentified as bad nodes during IDS voting especially if auditing is not performed. This is modeled by moving a good node in place N_g to place N_e after firing transition $T_{falsePositive}$ with a rate of $N_g \times P_{fp}^{IDS}/T_{IDS}$ where $P_{fp}^{IDS}$is the system-level false positive probability as a result of IDS voting (as given in Equation (6)) and T_{IDS} is the intrusion detection interval. The transition rate is set in this way because $1/T_{IDS}$ is the rate at which IDS voting is performed and each good node has a probability of P_{fp}^{IDS} to be misidentified as a bad node. Since we have a total of N_g good nodes, we multiply the per-node false positive rate with N_g to get the aggregate rate for transition $T_{falsePositive}$.
- When a bad node is being evaluated by IDS voting, if the voting outcome is negative (that is, the majority vote is no) then the bad node is evicted from the system. This corresponds to the case in which the system correctly detects the bad node with probability $1 - P_{fn}^{IDS}$ where P_{fn}^{IDS} is the system-level false negative probability (as given in Equation (6)). We create a timed transition $T_{truePositive}$ to model this "true positive" case, with the transition rate assigned to $T_{truePositive}$ being $N_b \times \left(1 - P_{fn}^{IDS}\right)/T_{IDS}$. The transition rate is set in this way because $1/T_{IDS}$ is the rate at which IDS voting is performed and each bad node has a probability of $1 - P_{fn}^{IDS}$ to be correctly identified as a bad node. Since we have a total of N_b bad nodes, we multiply the per-node true positive rate with N_b to get the aggregate rate for transition $T_{truePositive}$.

$$P_{fp}^{IDS}\ or\ P_{fn}^{IDS} = \sum_{i=0}^{m-m_{maj}} \left[\frac{C\binom{N_{bad}^{a}}{m_{maj}+i} \times C\binom{N_g + N_{bad}^{i}}{m-(m_{maj}+i)}}{C\binom{N_{bad}^{a}+N_{bad}^{i}+N_g}{m}} \right] + \sum_{i=0}^{m-m_{maj}} \left[\frac{C\binom{N_{bad}^{a}}{i} \times \sum_{j=m_{maj}-i}^{m-i} \left[C\binom{N_g + N_{bad}^{i}}{j} \times \omega^{j} \times C\binom{N_g + N_{bad}^{i} - j}{m-i-j} \times (1-\omega)^{m-i-j} \right]}{C\binom{N_{bad}^{a}+N_{bad}^{i}+N_g}{m}} \right] \quad (6)$$

- If the system misidentifies a bad node as a good node, then the bad node will remain in the system. We create a timed transition $T_{falseNegative}$ to model this "false negative" case, with the aggregate transition rate assigned to $T_{falseNegative}$ being $N_b \times P_{fn}^{IDS}/T_{IDS}$. The transition rate is set in this way because $1/T_{IDS}$ is the rate at which IDS voting is performed and each bad node has a probability of P_{fn}^{IDS} to be misidentified as a good node. Since we have a total of N_b bad nodes, we multiply the per-node false negative rate with N_b to get the aggregate rate for transition $T_{falseNegative}$. All such "false negative" bad nodes flow to a temporary place holder (the place that does not have a label in Figure 1) waiting to be distributed depending on the attack-defense conditions during IDS voting.
- The joint probability that a bad node attacks and the defense system audits during IDS voting is $P_a P_c$. If a system auditing is performed, the defense system will discover that there is a mismatch between the vote cast by the bad node and the auditing outcome. Consequently, a reduction of β life quota will be applied to the bad node to penalize this detected attack behavior during IDS voting. Since β=1 in Figure 1, a bad node in this case will lose its entire life quota and will be evicted, i.e., a bad node will flow to place N_e. We model this behavior by creating two "immediate" transitions (represented by two solid bars in Figure 1) with probabilities $P_a P_c$ and $1 - P_a P_c$, allowing a "false negative" bad node held in the temporary place holder to flow to N_e and N_b, respectively. In the underlying Markov model generated from the SPN model, a bad node will go directly from N_b to N_e if it decides to attack during an IDS cycle and the defense system also decides to audit in the same IDS cycle, and will remain in N_b in all other conditions.

The intrusion detection capability of our proposed IDS voting game is measured by the system-level false positive probability P_{fp}^{IDS} and the system-level false negative probability P_{fn}^{IDS} which in turn depend on the intrusion detection capability of individual nodes measured by the host-level false negative probability H_{pfn} and false positive probability H_{pfp} as well as the number of bad nodes performing attacks during IDS voting. Note that the system-level false positive probability P_{fp}^{IDS} is different from the host-level false positive probability H_{pfp} with the former being the result of IDS voting and the latter being the basic per-host detection capability of each individual node when a node is manufactured and put into operational use after a testing phase. Equation (6) derives the false positive probability (P_{fp}^{IDS}) and false negative probability (P_{fn}^{IDS}) when there are N_g good nodes and N_b bad nodes in the system. Equation (6) gives a closed-form solution for P_{fp}^{IDS} and P_{fn}^{IDS} under random attack behavior where $N_{bad}^{a} = P_a N_b$ and $N_{bad}^{i} = (1 - P_a)N_b$ are the numbers of "active" and "inactive" bad nodes, respectively; m_{maj} is the minimum majority of the number of voting nodes (m), e.g., 3 is the minimum majority of 5; ω is H_{pfp} for calculating P_{fp}^{IDS} and is H_{pfn} for calculating P_{fn}^{IDS}.

We explain Equation (6) for the system-level false positive probability P_{fp}^{IDS} below. The explanation to the system-level false negative probability P_{fn}^{IDS} is similar. A false positive will result when the majority vote is "no" against the target node (which is a good node). The first term in Equation (6) accounts for the case in which more than 1/2 of the voters selected from the target node's neighbors are "active" bad nodes who, as a result of actively performing attacks, will always vote against a good node as a bad node. Since more than 1/2 of the m voters vote no, the target node (which is a good node) is diagnosed as a bad node in this case, resulting in a false positive. Here the denominator is the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select at least m_{maj} bad voters out of N_{bad}^{a} nodes and the remaining good voters out of $N_g + N_{bad}^{i}$ nodes.

The second term accounts for the case in which more than 1/2 of the voters selected from the neighbors are good nodes but unfortunately some of these good nodes mistakenly misidentify the target nodes as a bad node with host IDS false positive probability $H_{pfp,}$ resulting in more than 1/2 of the voters (although some of those are good nodes) voting to evict the good target node. Since more than 1/2 of the m voters vote to evict, the target node (which is a good node) is also diagnosed as a bad node in this case, again resulting in a false positive. Here the denominator is again the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select i "active" bad voters not exceeding the majority $m_{maj,}$ j good or "inactive" bad voters who diagnose incorrectly with $i + j \geq m_{maj}$, and the remaining $m - i - j$ good or "inactive" voters who diagnose correctly. Here we note that an "inactive" bad node acts as if it is a good node based on our IDS attack-defense game setting.

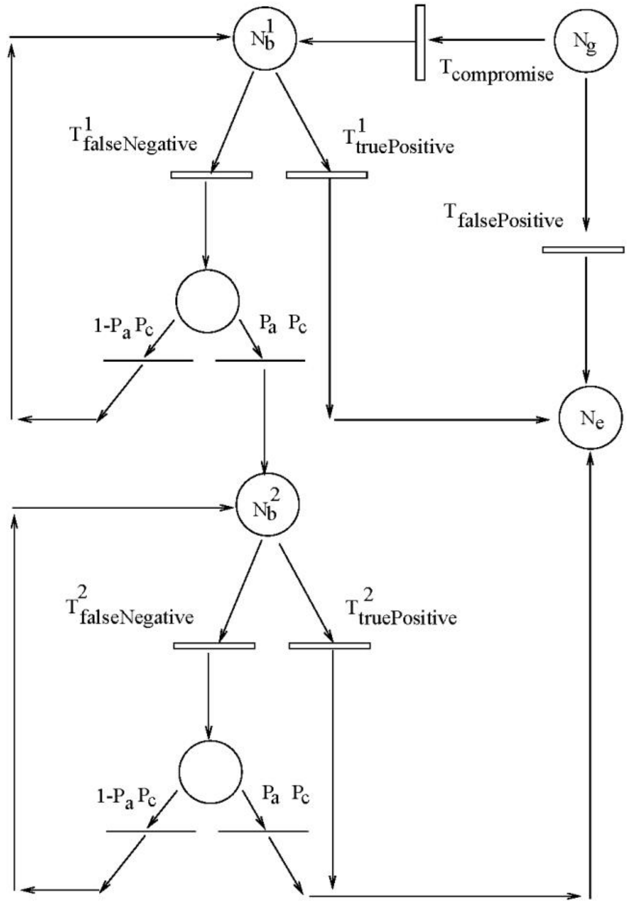

Figure 2: SPN Model for β=1/2.

The SPN model shown in Figure 1 models the case in which β=1 meaning that a single offense by a bad node during an IDS voting cycle will result in the bad node being evicted because it will lose its entire life quota. The SPN model can be easily extended to other cases. Figure 2 shows another SPN model for modeling the case in which β=1/2 meaning that on the first offense, a bad node will lose 1/2 of its life quota but is still allowed to remain in the system. However, on the second offense, a bad node will lose its entire life quota and will be evicted from the system. The SPN model in Figure 2 looks similar in structure to the SPN model in Figure 1. The upper layer is exactly the same except that the output place for the right immediate transition (with probability P_aP_c) is N_b^2 which is a new place created to hold bad nodes with a life quota of 1/2 (hence a superscript of "2" on N_b), because when β=1/2, a bad node will not lose all its entire life quota upon the first offense and instead will remain in the system with 1/2 life quota. The lower layer is a mirror image of the SPN model in Figure 1, except that a superscript of 2 is being used to denote that all bad nodes in the lower layer have only 1/2 life quota left.

The SPN model development can be generalized as follows: If β=$1/n$ then there will be n layers in the SPN model, i.e., layers *1, 2, …, n,* modeling the behaviors of bad nodes with life quota of *n/n, (n-1)/n, (n-2)/n, …, 1/n*, respectively. For example, Figure 1 for β=*1/1* only has one layer for modeling bad nodes with life quota of *1/1*, and Figure 2 for β=*1/2* has two layers for modeling bad nodes with life quota of *2/2*, and *1/2*, respectively.

V. Results

We apply our IDS attack-defense game design to an autonomous mobile CPS [11] comprising various types of IoT devices, including sensor-carried human actors, vehicles, and robots, assembled together for executing a mission in battlefield or emergency response situations. We setup the testing environment conditions and IDS attack-defense strategies as follows:

- The team consists of *N*=128 nodes moving randomly in 5×5 operational locations, with each location covering *R*=250m radio range based on 802.11n. Nodes that are in the same location at time *t* are considered neighbors at time *t*.
- All nodes have an equal chance to be captured by outside attackers or virus attacks and then will be compromised into malicious nodes. The per-node capture rate is λ. Once a node is compromised, it becomes an inside attacker and performs attacks with probability P_a whenever participating in IDS voting. In the experiment, we vary P_a to test its effect on performance.
- IDS voting is performed periodically in every T_{IDS} interval with m being the number of neighboring nodes to perform majority voting (toward a target node).
- We follow the energy model of [11] to consider the cost of each IDS voting cycle as well as the cost of each defense system audit (the *C* term in Table I). We consider that every node in the system is being voted on by m other nodes during an IDS voting cycle. This leads to an estimate of the overall energy consumption in each IDS voting cycle. The cost of each audit depends on the number of nodes being contacted to provide evidence to the defense system to audit the voting outcome. For the baseline mobile CPS [11], we assume that one half of all nodes are being contacted to provide evidence. This leads to an estimate of the overall energy consumption in each audit operation performed by the defense system. Due to the high cost of auditing, the defense system only performs it occasionally with probability P_c.
- In the experiment, we vary the defense system auditing probability P_c to test its effect on performance. The minimum auditing probability P_c^{min} is set to be $1/(1+\beta)$ for satisfying Condition (5).
- Each node is equipped with a host-level anomaly-based intrusion detection system characterized by a false negative probability H_{pfn} and a false positive

probability H_{pfp}.

- Byzantine failure [20] or energy depletion failure (as discussed in Section II.A) will cause the autonomous IoT system to fail.

Table II: Attack-Defense Parameters for a Baseline IoT System.

Parameter	Meaning	Default Value/Range
N	Number of nodes	128
H_{pfn}	False negative probability	5%
H_{pfp}	False positive probability	5%
λ	Per-node capture rate	1/7200 – 1/600
m	Number of voters	3, 5, 7
T_{IDS}	IDS interval	0 – 1400 time units
β	Life quota decay parameter	1/3, 1/2, 1
P_c^{min}	Minimum $P_c = 1/(1+\beta)$	3/4, 2/3, 1/2
P_c	Auditing probability	0, 0.25, 0.5, 0.75, 1
P_a	Attack probability	0, 0.25, 0.5, 0.75, 1

Table II lists the attack-defense strategy parameters for this autonomous collaborative IoT system. The performance metric is the system reliability expressed in terms of MTTF (mean time to failure). We obtain numerical results by parameterizing model parameters of the SPN model in Figure 1 (when $\beta = 1$) and Figure 2 (when $\beta = 1/2$) and running the SPN model through the SPNP tool [15] to obtain MTTF as the output.

Figure 3 shows the effect of T_{IDS} (X coordinate) on MTTF (Y coordinate) with varying attack probability P_a for the case in which β=1 and consequently the minimum auditing probability $P_c^{min} = 1/(1+\beta) = 0.5$ in our IDS attack-defense game.

We first observe that an optimal T_{IDS} (the IDS detection interval) exists at which the MTTF is maximized to best trade energy consumption for defense strength. When T_{IDS} is too small, the system performs intrusion detection too frequently and quickly exhausts its energy, thus resulting in a small lifetime. As T_{IDS} increases, the system saves more energy and its lifetime increases. On the other hand, when T_{IDS} is too large, even although the system can save more energy, it fails to catch bad nodes often enough, resulting in the system having many bad nodes. When the system has 1/3 or more bad nodes out of the total population, a Byzantine failure occurs. We also notice that optimal T_{IDS} value decreases as the attack probability P_a increases. The reason is that as the attack probability P_a increases, the system must perform IDS voting more often to more quickly remove malicious nodes to prevent them from attacking during IDS voting and evicting good target nodes, thus preventing Byzantine failures from occurring.

Here we also note that based on our IDS game design, since we set the defense system's auditing probability $P_c = P_c^{min} = 1/(1+\beta) = 0.5$, malicious nodes will not have incentives to attack because otherwise the payoff is less than zero. This corresponds to the case $P_a = 0$ at which the system has the highest MTTF, as shown in Figure 3. The reason is that when bad nodes do not attack during IDS voting, the system is less likely to experience Byzantine failures since good nodes would be preserved and bad nodes would be evicted based on the host-level intrusion detection capability as prescribed by Equation (6). Our IDS attack-defense game effectively discourages bad nodes from attacking and greatly improves the system MTTF.

Figure 4 shows the effect of P_c (X coordinate) on MTTF (Y coordinate) with varying attack probability P_a again for the case in which β=1 and consequently the minimum auditing probability $P_c^{min} = 1/(1+\beta) = 0.5$ in our IDS attack-defense game.

Two special cases are especially of interest:

- $P_c = 0.5$: The MTTF value represents the MTTF obtainable by our IDS game because following our game design, the defense system sets the auditing probability as $P_c^{min} = 1/(1+\beta) = 0.5$ to satisfy Condition (5). Particularly, when bad nodes are discouraged from performing attack during IDS voting, i.e., $P_a = 0$, the MTTF value is the highest MTTF value one can best hope for.
- $P_c = 0$: This is the "no auditing" case in which the defense system does not audit at all. The MTTF value represents the MTTF value obtainable by the baseline mobile CPS [11], which we will use for performance comparison.

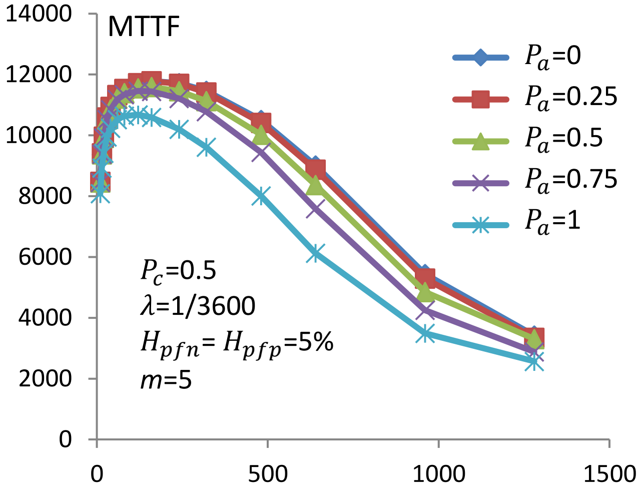

Figure 3: Effect of T_{IDS} on MTTF under varying attack probability P_a. Given P_a, there exists an optimal T_{IDS} value for maximizing MTTF.

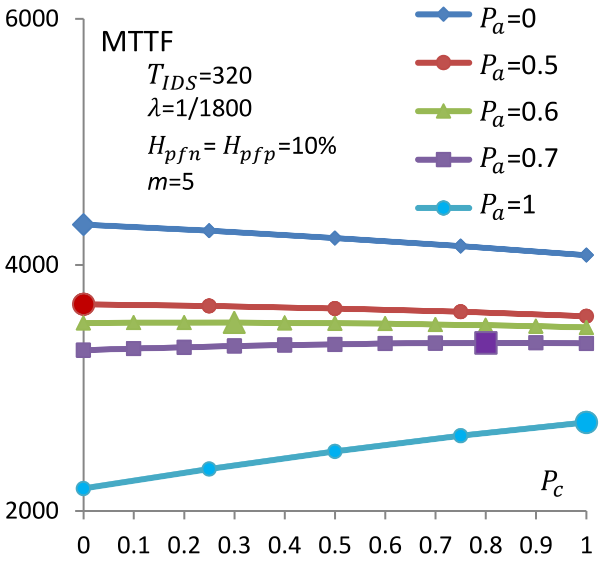

Figure 4: Effect of P_c on MTTF under varying attack probability P_a. Given P_a, there exists an optimal P_c value for maximizing MTTF.

We next analyze the effect of the auditing probability. Figure 4 shows that as P_a (the defense system auditing probability) increases the system MTTF decreases, except for the special case $P_a = 0$ in which the bad nodes do not attack because the system wastes energy for performing auditing. Further, given a P_a value, there is an optimal P_c value that maximizes the system MTTF. For example, the optimal P_c values are 0, 0.3, 0.8, and 1 (marked with bold phase) when $P_a \leq 0.5$, $P_a = 0.6$, $P_a = 0.7$, and $P_a = 1$, respectively. When P_a is low, the disadvantage of auditing (wasting energy) outweighs the advantage of auditing (evicting bad nodes and preserving good nodes). Consequently, the optimal P_c value is low. Conversely, when P_a is high, the advantage of auditing (evicting bad nodes and preserving good nodes) outweighs the disadvantage of auditing (wasting energy) and the optimal P_c value is high. If bad nodes attack all the time, i.e., $P_a = 1$, the system is better off by performing auditing on every IDS voting outcome. In this case the system can best prolong the system lifetime by delaying Byzantine failure from happening at the expense of inducing energy depletion failure caused by energy consumption due to frequent auditing.

Figure 5 compares our IDS attack-defense game against the baseline mobile CPS without IDS game design [11] head-to-head in terms of percentage gain or loss in MTTF. The MTTF loss happens when P_a is low ($P_a \leq 0.5$) due to energy wasted for excessively auditing the IDS voting outcomes. On the other hand, the MTTF gain happens when P_a is high ($P_a > 0.5$) due to timely removal of malicious nodes who decide to perform attacks during IDS voting. The exact point at which the tradeoff occurs depends on the magnitude of the cost per auditing (the C term in the IDS game). In case C is low, our IDS game will always gain in MTTF when compared to the baseline mobile CPS because of little risk of increasing the probability of energy depletion failure due to auditing. In our experiment setup, we set a high C value in order to realistically reflect the high cost associated with each audit, i.e., half of the 128 nodes are involved in providing evidence to the defense system to audit the IDS voting outcome.

As shown in Figure 5, when P_a is high, our design outperforms the baseline system [11] and the gain of MTTF is more significant as the auditing probability P_c increases because when bad nodes attack often during IDS voting, the risk of Byzantine failure is higher, so the system MTTF is higher by auditing more frequently to prevent good nodes from being removed and bad nodes from being retained. On the other hand, when P_a is low our design performs worse than the baseline system [11] especially as the auditing probability P_c increases because of unnecessary energy waste for performing auditing leading to a high risk of energy depletion failure.

It is noteworthy that the gain in MTTF (which happens when P_a is high) is large in magnitude relative to the loss in MTTF (which happens when P_a is low). This indicates that our IDS attack-defense game is most effective in highly hostile environments where attacks are reckless and intensive such that the attackers are eager to bring down the autonomous IoT system, so they perform attack whenever there is a chance. In this case, the system can best prolong the lifetime of the IoT system by performing auditing frequently even if the cost of auditing is high.

While Figure 5 shows the MTTF % gain/loss is a function of both P_a and P_c. In practice we do not know P_a. Therefore, it is not possible to dynamically set P_c to its optimal value to maximize the system MTTF. Following our IDS game design, the system designer should set P_c to $P_c^{min} = 1/(1+\beta) = 0.5$ to discourage bad nodes from performing attacks during IDS voting. There are two possibilities depending on how bad nodes react to the design that the defense system will audit 50% of the time:

- Bad nodes are sensible and they follow the payoff logic of our game design knowing that they should not attack because otherwise their payoff would be less than zero in which case the attack probability P_a is forced to set to zero at which the system would achieve its maximum achievable MTTF, as demonstrated in Figures 3 and 4.
- Bad nodes are not logical and they do not follow our IDS game design and still attack with their original attack probability P_a in which case we see from Figure 5 that under $P_c = 0.5$ the MTTF loss (which happens due to energy waste when P_a is low) is negligibly small in magnitude relative to the MTTF gain (which happens due to high detection rate when P_a is high). Therefore, even if bad nodes are not sensible, our IDS game design effectively improves the system MTTF by trading off energy (thus inducing energy depletion failure) for achieving a higher true positive rate and a lower false positive rate (thus delaying Byzantine failure), the effect of which is especially pro-

nounced in hostile environments where attacks are reckless and intensive.

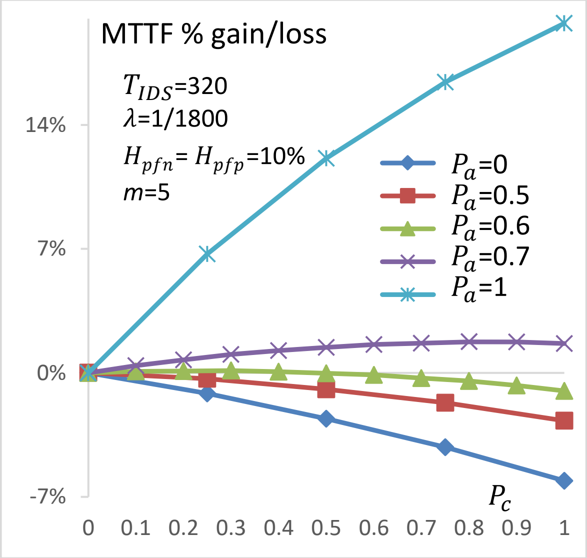

Figure 5: Performance comparison of our IDS game model vs. the baseline mobile CPS [11] in MTTF percentage gain/loss under varying attack probability P_a. Our design sets $P_c = 0.5$ to satisfy Condition (5) at which the MTTF gain (due to high detection rate when attack is intensive) outweighs the MTTF loss (due to energy waste when attack is not intensive).

VI. Conclusion and Future Work

In this paper we pioneered the concept of IDS attack-defense games to incentivize nodes to cooperate in executing intrusion detection with the objective to maximize the system reliability of autonomous IoT systems. We analytically derived the exact condition under which malicious nodes will not have the incentive to attack during IDS voting. We also developed an analytical model based on SPN modeling techniques to analyze the effects of attack-defense behaviors deriving from attack probability, defense audit probability, IDS strength, and voting outcome mismatch penalty on detection accuracy and system reliability. We illustrated the practical use of our IDS attack-defense game by applying it to a mobile CPS wherein each IoT device is given a life quota. The results demonstrate that our IDS attack-defense game design greatly improves system reliability over the baseline mobile CPS when malicious nodes are sensible. When malicious nodes are not sensible, our IDS game design effectively improves the system MTTF by trading off energy (thus inducing energy depletion failure) for achieving a higher detection rate (thus delaying Byzantine failure), especially for hostile environments where attacks are reckless and intensive.

The baseline autonomous IoT system considered in this paper for comparative performance analysis is a homogeneous mobile CPS, so a single SPN model can model the attack-defense behavior adequately. In the future we plan to apply our IDS game to autonomous IoT systems for which nodes are heterogeneous. This necessitates the use of a hierarchical SPN model with the low-level models describing diverse attack-defense behaviors and the upper-level models describing the aggregate behaviors and system responses.

Acknowledgment

This work is supported in part by the U.S. AFOSR under grant number FA2386-17-1-4076.

References

[1] J. Timpner, D. Schürmann, and L. Wolf, "Trustworthy parking communities: Helping your neighbor to find a space," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 1, pp. 120-132, 2016.

[2] J. Guo, I.R. Chen, D.C. Wang, J.J.P. Tsai, and H. Al-Hamadi, "Trust-based IoT Cloud Participatory Sensing of Air Quality," *Wireless Personal Communications*, vol. 105, no. 4, 2019, pp. 1461-1474.

[3] H. Al-Hamadi, I.R. Chen, and J.H. Cho, "Trust Management of Smart Service Communities," *IEEE Access*, vol. 7, no. 1, 2019, pp. 26362-26378.

[4] I.R. Chen, J. Guo, D.C. Wang, J.J.P. Tsai, H. Al-Hamadi, and I. You, "Trust-based Service Management for Mobile Cloud IoT Systems," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, 2019, pp. 246-263.

[5] L. Xiao, T. Chen, C. Xie, H. Dai, and H.V. Poor, "Mobile Crowdsensing Games in Vehicular Networks", *IEEE Transactions on Vehicular Technology*, vol. 67, no. 2, 2018, pp. 1535-1545.

[6] L. Pu et al., "Chimera: An Energy-Efficient and Deadline-Aware Hybrid Edge Computing Framework for Vehicular Crowdsensing Applications," *IEEE IoT Journal*, vol. 6, no. 1, 2019, pp. 84-99.

[7] L. Liang, H. Ye, and G.Y. Li, "Toward Intelligent Vehicular Networks: A Machine Learning Framework," *IEEE IoT Journal*, vol. 6, no. 1, 2019, pp. 124-135.

[8] I. You, K. Yim, V. Sharma, I.R. Chen, and J.H. Cho, "Misbehavior Detection of Embedded IoT Devices in Medical Cyber Physical Systems," *3rd ACM Chase Workshop on Security, Privacy, and Trustworthiness in Medical Cyber-Physical Systems*, Washington DC, Sept 2018.

[9] I. You, K. Yim, V. Sharma, I.R. Chen, and J.H. Cho, "On IoT Misbehavior Detection in Cyber Physical Systems, *23rd IEEE Pacific Rim International Symposium on Dependable Computing*, Taipei, Dec 2018.

[10] H. Al-Hamadi and I.R. Chen, "Trust-Based Decision Making for Health IoT Systems," *IEEE Internet of Things Journal*, vol. 4, no. 5, Oct. 2017, pp. 1408-1419.

[11] R. Mitchell and I.R. Chen, "On Survivability of Mobile Cyber Physical Systems with Intrusion Detection," *Wireless Personal Communications*, vol. 68, no. 4, 2013, pp. 1377-1391.

[12] L. Hurwicz and S. Reiter, *Designing Economic Mechanisms, Cambridge University Press*, 2006.

[13] L. Canzian, Y. Xiao, W. Zame, M. Zorzi, and M. van der Schaar,

``Intervention with private information, imperfect monitoring and costly communication," *IEEE Transactions on Communications*, vol. 61, no. 8, pp. 3192--3205, 2013.

[14] J. Wang, I.R. Chen, J.J.P. Tsai, and D.C. Wang, "Trust-based Cooperative Spectrum Sensing in Cognitive Radio Networks," *Computer Communications*, vol. 116, 2018, pp. 90-100.

[15] G. Ciardo, R.M. Fricks, J.K. Muppala, and K.S. Trivedi, *Stochastic Petri Net Package (SPNP)*, Duke University, 1999.

[16] I.R. Chen and D.C. Wang, "Analyzing dynamic voting using Petri nets," *15th Symposium on Reliable Distributed Systems*, 1996, pp. 44-53.

[17] I.R. Chen and T.H. Hsi, "Performance analysis of admission control algorithms based on reward optimization for real-time multimedia servers," *Performance Evaluation*, vol. 33, no. 2, 1998, pp. 89-112.

[18] S.T. Cheng, C.M. Chen, and I.R. Chen, "Dynamic quota-based admission control with sub-rating in multimedia servers," *Multimedia Systems*, vol. 8, no. 2, 2000, pp. 83-91.

[19] F.B. Bastani, I.R. Chen, and T. Tsao, "Reliability of Systems with Fuzzy-Failure Criterion," *Annual Reliability and Maintainability Symposium*, 1994, pp. 442-448.

[20] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, 1982, pp. 382-401.

AUTHOR BIOGRAPHIES

Ding-Chau Wang received the BS degree from Tung-Hai University, Taichung, Taiwan, and the MS and PhD degrees in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan. He is currently an associate professor in the Department of Information Management at Southern Taiwan University of Science and Technology, Tainan, Taiwan. His research interests include game-based learning, Internet of things, mobile computing, security, database systems and performance analysis.

Ing-Ray Chen received the BS degree from the National Taiwan University, and the MS and PhD degrees in computer science from the University of Houston. He is a professor in the Department of Computer Science at Virginia Tech. His research interests include mobile computing, wireless systems, security, trust management, and reliability and performance analysis. Dr. Chen currently serves as an editor for *IEEE Transactions on Services Computing*, *IEEE Transactions on Network and Service Management*, and *The Computer Journal*. He is a recipient of the IEEE Communications Society William R. Bennett Prize in the field of Communications Networking.

Hamid Al-Hamadi received the B.S. degree in information technology from Griffith University, Brisbane, Australia, in 2003 and the M.S. degree in information technology from the Queensland University of Technology, Brisbane, Australia, in 2005 and the Ph.D. degree in computer science from the Virginia Polytechnic Institute and State University, VA, USA in 2014. He has experience working as a Network Engineer at Kuwait National Petroleum Company and at Tawasul Telecom, Kuwait. Currently, he is an Assistant Professor with the Department of Computer Science, Kuwait University, Kuwait. His current research interests include Internet of Things, security, mobile cloud, trust management, and reliability and performance analysis.