A Node and Load Allocation Algorithm for Resilient CPSs under Energy-Exhaustion Attack

Thidapat Chantem
Electrical and Computer Engineering
Utah State University
Logan, UT 84322, USA
Email: tam.chantem@usu.edu

Ryan M. Gerdes
Electrical and Computer Engineering
Utah State University
Logan, UT 84322, USA
Email: tam.chantem@usu.edu

Abstract—**A cyber-physical system (CPS) must continue to meet minimum performance requirements under extreme conditions, such as in the presence of security breaches. Existing security-aware techniques and analyses focus on pre-attack countermeasures and do not consider the impact of security attacks on real-time performance. We propose a post-attack node and load allocation algorithm for CPSs that must continue operation even under energy-exhaustion attack. Our algorithm is designed based on probabilistic programming where chance constraints are used to capture the potential effects of an attack, thus allowing the CPSs to be more resilient. Simulation results indicate that our method is able to significantly increase the remaining CPS energy, thus extending the operating lifetime of the CPS.**

I. Introduction

A cyber-physical system (CPS) is a system in which the physical component (e.g., sensors and actuators) is tightly coupled with the computation and communication components (e.g., smart control and intelligent data fusion). Efficiency, dependability, security, and timeliness are the primary concerns when designing a CPS. For example, an outdoor tactical border surveillance system consisting of batteried nodes that detect motion and capture images must be efficient since its primary energy source may be solar. At the same time, the system must be available when needed, secure enough that it is reasonably hard to compromise, able to deliver data in a timely manner so that appropriate actions can be taken, and able to response to changes in environments or application requirements.

The availability of a potentially large number of nodes, i.e., processors or other computing devices, in a CPS allows for a natural design of an energy-efficient system. Namely, a CPS may only make use of a subset of its nodes to save energy and allow other nodes an opportunity for battery recharges. In this paper, we will focus primarily on the energy-efficiency and security aspects of a CPS and their relationships to real-time performance. Several research papers have addressed various security aspects in the context of CPSs, e.g., [1], [13], [14], [16], [17] and real-time systems, e.g., [4], [5], [7], [15]. However, an unexplored area of research is providing CPS livelihood resilience post attack. In this work, we focus on security breaches that seek to degrade CPS performance by draining the energy resources of individual nodes [13]. In these so-called battery or energy-exhaustion attacks [3], [13], an adversary interacts with a node in legitimate ways but with ill-intent so as to prevent the node from going into a lower energy consumption sleep state. This "sleep deprivation torture" affects the availability of the node through an increased draining of its energy supply that may force a premature shutdown or degraded performance. In this work, we generalize the aforementioned energy attacks in that a node may not only be forced to remain awake, but that it may need to operate in a higher energy consumption mode due to increased in its workload. Using our previous example, it is not hard to imagine that the outdoor tactical border surveillance system may be a victim of energy exhaustion attacks so that the surveillance system provides incomplete coverage. In such a case, the desirable behavior of the system would be to continue its operation even when under attack while providing the best possible coverage.

To jointly optimize for CPS energy efficiency and resilience, we consider CPSs consisting of a number of nodes, some of which may be under attack and forced to expend energy. The CPS must decide (i) which nodes

978-1-4799-6177-1/14/$31.00 ©2014 IEEE

to assign loads to, and (ii) how much workload, i.e., utilization to give to each node in order to meet a specific performance metric and save the total remaining system energy to increase CPS lifetime. This *node and load* allocation problem is first solved using a chance constrained programming formulation [10], [11], to capture the uncertainty due to an attack. For run-time use, we propose an efficient heuristic algorithm that considers the impact of energy attacks on a node's energy state when performing node and load allocation.

II. Preliminaries

We present our system model and state our assumptions in this section.

A. Node Model

We consider a CPS with $|M|$ heterogeneous nodes. Different nodes may execute different sets of periodic real-time tasks. The actual utilization, i.e., amount of work to be completed over time, of a node m_i is denoted U_i, $i = 1, \ldots, |M|$, and is to be determined during the node and load allocation process. We assume that the Earliest Deadline First (EDF) scheduling algorithm [6] is used to schedule tasks on each node.

Each node m_i runs on a battery with energy harvesting capability. At time t_0, the energy level of a node is denoted as E_{i,t_0}. The energy consumption by m_i during time interval $[t_0, t]$, $t \geq t_0$ is

$$E_{U_i,t-t_0} = (\alpha_i + P_i)(t - t_0), \tag{1}$$

where α_i is a constant associated with static power consumption of node m_i and $P_i = C_{ef} V_{dd}^2 s$, where $s = \kappa \frac{(V_{dd}-V_t)^2}{V_{dd}}$, and C_{ef}, V_t, V_{dd}, and κ represent the effective switch capacitance, the threshold voltage, the supply voltage, and a hardware-design-specific constant, respectively [2], [12]. Each node is equipped with dynamic voltage and frequency scaling (DVFS) capability. The actual voltage and frequency pair used by m_i is referred to collectively as speed level $s_i \in [0, 1]$.

During the time interval $[t_0, t]$, a node may be under energy attack. The energy wasted due to such an attack can be expressed as

$$\mathring{E}_{i,t-t_0} = \mathring{P}_i \cdot (t - t_0), \tag{2}$$

where $\mathring{P}_i$ is the power consumption due to the attack, $i = 1, \ldots, |M|$ and which may not be precisely known but can be predicted using models or via profiling [8]. Given the current time t_0, the total remaining energy of m_i at time t, $t \geq t_0$ is

$$E_{i,t} = E_{i,t_0} - E_{U_i,t-t_0} - \mathring{E}_{i,t-t_0} + E^r_{i,t-t_0}, \tag{3}$$

where $E^r_{i,t-t_0}$ is the harvest energy during that time interval. The total remaining system energy is

$$E_{sys,t} = \sum_{i=1}^{|M|} E_{i,t}. \tag{4}$$

B. Energy Attack Model

We will use the work by Mitchell and Chen [8] to identify nodes in a CPS as compromised (under attack) or non-compromised (free from attack), with a given false-positive and false-negative rate, based on the degree of compliance to an ideal node's behavior [8]. We assume insidious attacks where the attacker perpetrates the attack against nodes at same time and for the same duration. The effect of the energy attack, carried out via extraneous interactions, is to increase the speed level of the node, i.e., the node is forced to execute at a higher voltage and frequency pair than the real-time task set assigned to it dictates. In other words, the total node utilization is increased in an effort to waste the node's precious resources. Obviously, the higher the speed level, the more energy consumed. Let s_i be the speed level needed to finish the workload assigned to a node by some deadlines as stated earlier, we assume that the impact of an attack leads to an increase in speed level of s'_i where $s'_i \in [1/3, 2/3]$. We further assume that the resultant speed level $s_i + s'_i$ can be determined with 5% accuracy [9].

III. Node and Load Allocation

In an energy attack, the main objective is to cause the nodes in the CPS to waste energy. Consequences of this type of attack include, but are not limited to, temporal overloads, excessive energy consumption, decreased performance, deadline misses, and dead nodes. Needless to say, an energy attack is especially effective in a wireless, batteried CPS, which is the focus of this work. Even if an attack can be detected, it may be difficult or time consuming for the system to respond with countermeasures. For these reasons, we propose a proactive approach to increase CPS resilience, as discussed next.

A. Problem Formulation

We will use mathematical programming to guide us in the design of a resilient node allocation algorithm.

However, simply formulating a node and load allocation problem as a standard constrained optimization problem is not practical since an energy attack may not be detected immediately or at all, especially if it is of lower impact. Also, the node must necessarily estimate the impact of an attack, which may not be precisely known. Both of these difficulties make the node and load allocation problem very difficult and a straightforward application of constrained optimization may result in suboptimal or even infeasible solutions. To overcome these challenges, we propose using chance constrained programming. Specifically, the node and load allocation problem can be formulated as follows.

$$\max: \quad E_{sys,t} = \sum_{i=1}^{|M|} E_{i,t} \tag{5}$$

$$\text{s.t.}: \quad \sum_{i=1}^{|M|} x_i U_i = U_{total} \tag{6}$$

$$0 \leq x_i U_i \leq 1 \tag{7}$$

$$\begin{aligned} &\mathbf{Pr}(E_{i,t} \geq \max\{E_{i,t_0} - x_i E_{U_i,t-t_0} - \\ &\quad x_i \xi_i \mathring{P}_i \cdot (t-t_0) + E^r_{i,t-t_0}, 0\}) \\ &\quad \geq 1-\alpha,\ i = 1, \ldots, |M| \end{aligned} \tag{8}$$

$$x \in \{0,1\} \tag{9}$$

where α is the risk level and is usually smaller than 0.05, U_{total} is the required total system utilization to maintain minimum performance level, t_0 is the current time and t is some future time point ($t \geq t_0$), $\xi_{i,q}$, $i = 1, \ldots, |M|$, denote the probability that an attack has been made on node m_i. The binary decision variable $x_i = 1$ if node m_i will be assigned a utilization U_i, and $x_i = 0$ otherwise, for $i = 1, \ldots, |M|$. The other variables are as defined in Section II-A. The above chance constrained programming aims to maximize the total remaining energy of the CPS under consideration. The constraints are to ensure that the required work is completed in a timely manner (6), that no node is assigned more workload than it can handle (7), and that the probability that a node's energy level is at least what is predicted is no less than as dictated by the risk level α (8). Note that we choose to optimize the total remaining system energy instead of actual node lifetimes since the latter are harder to compute when the workload allocated to a given node may frequently change, as is in our case.

The chance constrained programming problem in (5)–(9) is a probabilistic formulation of a variation of the knapsack problem, which is known to be NP-hard even in the deterministic case. Combined with the fact that the majority of chance constrained programming problems cannot be efficiently solved at runtime, we propose using an efficient heuristic to solve our problem.

B. *Efficient Heuristic*

We begin by defining the *relative energy index* $\hat{E}_{i,t-t_0}$ of a given node m_i during the time duration $t - t_0$ as the ratio of the sum of the energy consumption required to execute a real-time task set at some required utilization level U_i and the energy expended by the node due to attacks over the remaining energy level and the energy gained due to recharging. In other words,

$$\hat{E}_{i,t-t_0} = \frac{\bar{P}_i \cdot (t-t_0) + E_{U_i,t-t_0}}{E_{i,t_0} + E^r_{i,t-t_0}}, \tag{10}$$

where $\bar{P}_i$, $i = 1, \ldots, |M|$, denote the **predicted** power consumption due to the attack. Note that the predicted power consumption value is used instead of the actual power consumption, as the definite impact of an energy attack may not be known until later, if at all. It is clear from (10) that a node with a lower value of $\hat{E}_{i,t-t_0}$ makes a relatively more efficient use of its resources. The concept of relative energy index not only allows for nodes to be ordered in some manner, but is general enough to be used in real CPSs where nodes may be heterogeneous in terms of computational power and recharge characteristics.

The uncertainties in the system stem from the probability that an attack is being carried out, as well as the impact of that attack on a node's energy level. If an attack has been made on node m_i but not detected, i.e., we have a false negative, the node's energy level will be underestimated, possibly resulting in the node running out of energy earlier than expected. On the other hand, if there is no attack but a detection has been made, i.e., we have a false positive, the node's energy level will be overestimated, possibly resulting in necessarily reduced performance. Clearly, a false negative has a bigger impact on a node's energy level than a false positive.

We now present our energy-aware node and load allocation algorithm, which is summarized in Alg. 1. Alg. 1 takes as inputs the set of nodes in a CPS, the total required system utilization U_{total}, the current time t_0, a future time point t where loads may be reallocated, and U_{step}, which is a user-defined parameter. The output, if any, is a set of nodes that must be online or available along with their assigned workload, i.e., utilization.

The main idea behind the proposed algorithm is to sort the nodes in a non-increasing order of relative energy index (Lines 6–8). The utilization level for each node is assigned iteratively. In each iteration, U_{step} is assigned to the node with the smallest relative energy index, i.e., the node with the best energy efficiency (Lines 14–23). This process continues until all the required system utilization has been allocated. There are two scenarios that Alg. 1 can encounter when returning an empty set, i.e., no solution. In the first scenario, U_{total} is greater than $|M|$, which means that the total required system utilization exceeds the total utilization that the CPS can execute assuming that each node can only handle a utilization of up to 1 (Lines 1–2). In the second scenario, the total system energy at time t is less than zero, which implies that there is not enough energy left in the CPS for the requested total system utilization (Lines 19–20).

To analyze some properties of Alg. 1, we define the deterministic version of the problem in (5)–(9) as

$$\max: \quad E_{sys,t} = \sum_{i=1}^{|M|} E_{i,t} \tag{11}$$

$$\text{s.t.}: \quad \sum_{i=1}^{|M|} x_i U_i = U_{total} \tag{12}$$

$$0 \le x_i U_i \le 1 \tag{13}$$

$$E_{i,t} = \max\{E_{i,t_0} - x_i E_{U_i,t-t_0} - x_i \mathring{E}_{i,t-t_0} + E^r_{i,t-t_0}, 0\},\ i = 1,\dots,|M| \tag{14}$$

$$x \in \{0,1\} \tag{15}$$

Some relevant properties of Alg. 1 are discussed in the following lemmas with proofs omitted to save space.

Lemma 1: A non-empty solution returned by Alg. 1 satisfies the constraints in (12), (13), (14), and (15).

Lemma 2: As the user-defined parameter $U_{step} \to 0$, Alg. 1 always returns a solution, if one exists.

One question that may arise from Lemma 2 is how to set U_{step}. Clearly, it is desirable to set U_{step} to a relatively small value to increase the likelihood that a solution will be found. However, the smaller U_{step}, the larger the runtime overhead. The most time consuming part of Alg. 1 is the while loop spanning from Lines 5–23. In the worst case, the loop will be iterated $U_{iter} = \frac{U_{total}}{U_{step}}$ times. The operation that sorts nodes in a non-decreasing order of energy index (Line 8) is the most time consuming step inside the while loop. Hence, the time complexity of Alg. 1 is $O(U_{iter}|M|\log|M|)$, where $|M|$ is the number of nodes in the system. The user-defined parameter U_{step} should be set as small as possible while large enough for the overhead of the algorithm to be tolerable online. Finally, our algorithm can be applied to clusters of nodes if the CPS is very large.

Algorithm 1 Allocation(M, U_{total}, t_0, t, U_{step})

```
1:  if $|M| < U_{total}$ then // Assuming each node can handle a utilization of up to 1
2:      return $\emptyset$
3:  $U_i \leftarrow U_i^{attackExp}$, $i = 1,\dots,|M|$ // $U_i^{attackExp}$ is the expected impact of attack on system utilization
4:  $U_{curr} \leftarrow 0$
5:  while $U_{curr} < U_{total}$ do
6:      for each node $m_i \in M$ do
7:          compute $\hat{E}_{i,t-t_0}$ according to (10)
8:      $M \leftarrow$ Sort_Nodes_by_Energy_Index($M$)
9:      for $m_i \in M$ do
10:         $E_{i,t} \leftarrow E_{i,t_0} - E_{U_i+U_{step},t-t_0} - \mathring{E}_{i,t-t_0} + E^r_{i,t-t_0}$ // Check to see if this node will have enough energy
11:         if $E_{i,t} < 0$ then
12:             continue
13:         else if $U_i + U_{step} \ge 1$ or $U_{curr} + U_{step} > U_{total}$ then
14:             $U_{rem} \leftarrow \min(1 - U_i, U_{total} - U_{curr})$
15:             $U_{curr} \leftarrow U_{curr} + U_{rem}$
16:             $U_i \leftarrow U_i + U_{rem}$
17:             if $\mathrm{abs}(1 - U_i) < \epsilon$ then // This node is fully loaded so remove it from further consideration
18:                 $M \leftarrow M - m_i$
19:                 if $M = \emptyset$ and $U_{curr} < U_{total}$ then
20:                     return $\emptyset$
21:         else
22:             $U_{curr} \leftarrow U_{curr} + U_{step}$
23:             $U_i \leftarrow U_i + U_{step}$
24: return U
```

IV. Evaluation

We now assess the performance of our algorithm using simulations.

A. Simulation Setup

Due to the lack of existing algorithms that consider real-time performance in face of security breaches, we compare our algorithm against the following algorithms,

which focus on optimizing the quality-of-service of applications subject to energy constraints on the nodes. The algorithms, which are based on the Largest-Estimated-Utilization-First strategy [2], will be used to demonstrate that failure to consider the impacts of energy-exhaustion attacks can lead to significant reductions in the lifetime of a tactical border surveillance system. We did not compare our method against an algorithm that has perfect knowledge of the attacks, as any performance gain would be due to superior attack detection and does not offer insights on the performance of the proposed algorithm.
Algorithm A: Sort nodes with the largest remaining energy first (predicted energy expended due to attacks is ignored). Assign to each node the maximum utilization it can handle in sorted order.
Algorithm B: Sort nodes with the largest remaining energy first (predicted energy expended due to attacks is ignored). The node with the largest remaining energy is incrementally assigned a utilization amount of U_{step}. The node list is resorted and the process is repeated until all the required workload has been assigned.

Unless stated otherwise, there are 128 nodes in the CPS. The value U_{step} is set to 0.1. For the sake of simplicity, we assumed that all the nodes were initially fully charged. We further assumed that for each node, the maximum utilization is 1 since EDF is used to schedule tasks. The speed level due to an attack is obtained using a uniform distribution while the errors in estimating the actual speed levels are based on a normal distribution.

Nodes are compromised from the outset of the simulation and no new nodes are compromised for its duration. Following the work by Mitchell and Chen [8], let the random variable $X_b \sim \text{Beta}(\alpha = 1, \beta_{p_a})$ represent the degree of compliance of a compromised node with an attack probability of p_a and using a compliance threshold $th = 0.90$, the probability p_{tp} that a compromised node is identified as compromised (a true positive) is given by $p_{tp} = \mathbf{Pr}\,(X_b < th) = I_{th}(\alpha = 1, \beta_{pa})$, where $I_x(\alpha, \beta)$ is the cumulative distribution function of the Beta distribution. On the other hand, the probability that it is incorrectly identified as uncompromised (a false negative) is $p_{fn} = \mathbf{Pr}\,(X_b \geq th) = 1 - I_{th}(\alpha = 1, \beta_{pa})$. Letting the random variable $X_g \sim \text{Beta}(\alpha = 1, \beta_g)$ represent the degree of compliance of a non-compromised node with a compliance threshold of 0.9, the probabilities of a true negative and a false positive can be similarly calculated. Given the above, a node will be identified as compromised if $1 - X_{b/g} > 0.1$, where X_b is used if the node is compromised and X_g if it is not.

B. Results

We first assess the performance of the proposed algorithm relative to Alg. A and B under different ratios of compromised nodes (the number of compromised nodes over the total number of nodes in the CPS). The results are shown in Fig. 1(a) and 1(b). From Fig. 1(a), it is obvious that our method is able to significantly reduce the number of dead nodes, thus allowing the tactical border surveillance system to remain operational for longer while meeting its performance requirements. In fact, when the ratio of compromised nodes is 1, i.e., all the nodes in the tactical border surveillance system are compromised, applying our algorithm results in only about 18 dead nodes. In contrast, none of the 128 nodes remain alive when applying Alg. A or B. On average, our approach allows for about 86% more nodes to remain alive, which is a significant improvement. Fig. 1(b) depicts the remaining energy for the different algorithms. Our method allows for the CPS to have remaining energy after the simulation ends, whereas using Alg. A and B result in the entire CPS being down.

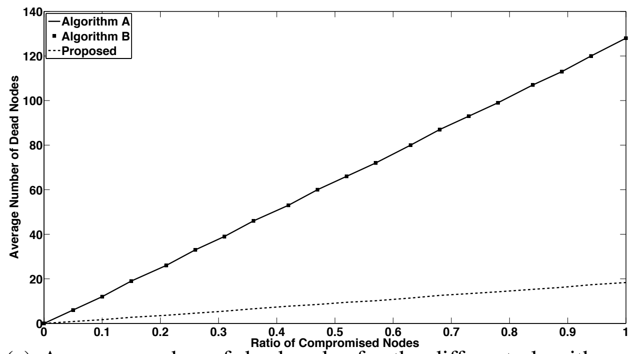
We now compare the performance of our algorithm against Alg. A and B for various CPS sizes, as shown in Fig. 2(a) and 2(b). For this set of simulations, the size of the tactical border surveillance system is between 10 and 5000. The ratio of compromised nodes is set to 0.25 and U_{step} set to 1 since, for large CPSs, the performance of Alg. B and 1 is most likely less sensitive to the value of U_{step}. From this set of simulations, it is clear that the proposed algorithm significantly outperforms the baseline algorithms. Specifically, Alg. 1 results in 98.9% more live nodes on average, and up to 99.7%, with positive remaining CPS energy.

V. Conclusions

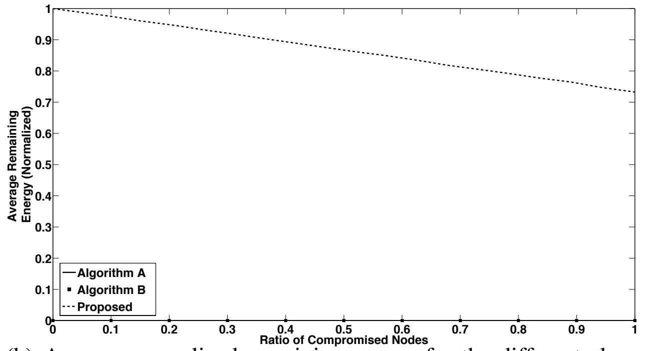
We showed that in the design of resilient CPS, even nodes that are under energy-exhaustion attack can be used to maintain the performance level of a CPS. The proposed node and load allocation algorithm thus allows for efficient use of precious resources and helps the system continue operation even in the presence of energy-exhaustion attack. As future work, we plan to consider other types of attacks in addition to energy-exhaustion attack and test our framework on real hardware platforms.

References

[1] A. A. Cardenas, S. Amin, and S. Sastry, "Secure control: Towards survivable cyber-physical systems," in *Proc. Int. Conf. Distributed Computing Systems Wksh.*, Jun. 2008, pp. 495–500.

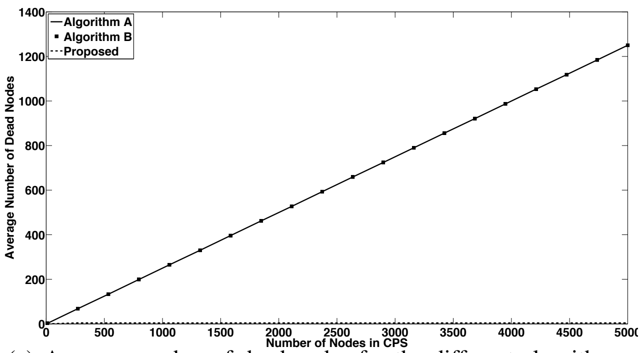


(a) Average number of dead nodes for the different algorithms.

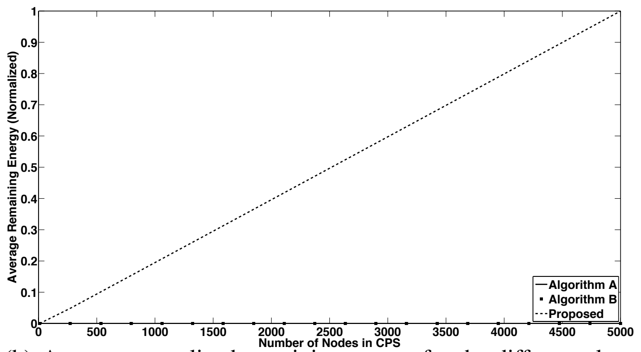


(b) Average normalized remaining energy for the different algorithms.

Fig. 1. Results for different ratios of compromised nodes.



(a) Average number of dead nodes for the different algorithms.



(b) Average normalized remaining energy for the different algorithms.

Fig. 2. Results for different tactical border surveillance system sizes.

[2] J.-J. Chen, C.-Y. Yang, H.-I. Lu, and T.-W. Kuo, "Approximation algorithms for multiprocessor energy-efficient scheduling of periodic real-time tasks with uncertain task execution time," in *Proc. Real-Time and Embedded Technology and Applications Symp.*, Apr. 2008, pp. 13–23.

[3] N. Gaddam, G. Kumar, and A. Somani, "Securing physical processes against cyber attacks in cyber-physical systems," in *Proc. National Wksh. for Research on High-Confidence Transportation Cyber-Physical Systems: Automotive, Aviation and Rail*, Nov. 2008.

[4] M. Lin, L. Xu, L. Yang, X. Qin, N. Zheng, Z. Wu, and M. Qiu, "Static security optimization for real-time systems," *IEEE Trans. Industrial Informatics*, vol. 5, no. 1, pp. 22–37, Feb. 2009.

[5] M. Lindberg and K. Årzén, "Feedback control of cyber-physical systems with multi resource dependencies and model uncertainties," in *Proc. Real-Time Systems Symp.*, Dec. 2010, pp. 85–94.

[6] C. Liu and J. Layland, "Scheduling algorithms for multiprogramming in a hard real-time environment," *J. of the ACM*, vol. 20, no. 1, pp. 46–61, Jan. 1973.

[7] Y. Ma, W. Jiang, N. Sang, and X. Zhang, "ARCSM: A distributed feedback control mechanism for security-critical real-time system," in *Proc. Int. Symp. Parallel and Distributed Processing with Applications*, Jul. 2012, pp. 379–386.

[8] R. Mitchell and I.-R. Chen, "Effect of intrusion detection and response on reliability of cyber physical systems," *IEEE Trans. Reliability*, vol. 62, no. 1, pp. 199–210, Mar. 2013.

[9] D. Nash, T. Martin, D. Ha, and M. Hsiao, "Towards an intrusion detection system for battery exhaustion attacks on mobile computing devices," in *Proc. Int. Conf. Pervasive Computing and Communications Wksh.*, Mar. 2005, pp. 141–145.

[10] A. Nemirovski and A. Shapiro, "Convex approximations of chance constrained programs," *SIAM J. Optimization*, vol. 17, no. 4, pp. 969–996, Dec. 2006.

[11] A. Prékopa, *Stochastic Programming*. Kluwer Academic Publishers, 1995.

[12] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits*, 2nd ed. Prentice Hall, 2002.

[13] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for ad-hoc wireless networks," in *Proc. Int. Wksh. Security Protocols*, Apr. 1999, pp. 172–194.

[14] E. Wang, Y. Ye, X. Xiaofei, S. Yiu, L. Hui, and K. Chow, "Security issues and challenges for cyber physical system," in *Proc. Int. Conf. Green Computing and Communications, and on Cyber, Physical and Social Computing*, Dec. 2010, pp. 733–738.

[15] T. Xie and X. Qin, "Scheduling security-critical real-time applications on clusters," *IEEE Trans. Computers*, vol. 55, no. 7, pp. 864–879, Jul. 2006.

[16] M. Yampolskiy, P. Horvath, X. Koutsoukos, Y. Xue, and J. Sztipanovits, "Systematic analysis of cyber-attacks on CPS - evaluating applicability of a DFD-based approach," in *Proc. of Int. Symp. Resilient Control Systems*, Aug. 2012.

[17] B. Zhu, A. Joseph, and S. Sastry, "A taxonomy of cyber attacks on SCADA systems," in *Proc. Int. Conf. Internet of Things, and on Cyber, Physical and Social Computing*, Oct. 2011, pp. 380–388.