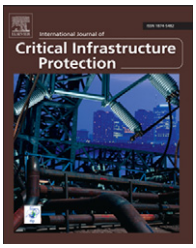
Available online at www.sciencedirect.com
SciVerse ScienceDirect
www.elsevier.com/locate/ijcip



Modeling security in cyber–physical systems

Mike Burmester[a,*], *Emmanouil Magkos*[b], *Vassilis Chrissikopoulos*[b]

[a]*Department of Computer Science, Florida State University, Tallahassee, Florida 32306-4530, USA*
[b]*Department of Informatics, Ionian University, Tsirigoti Square 7, 49100 Corfu, Greece*

ARTICLE INFO

Article history:
Received 20 May 2012
Available online 18 August 2012

Keywords:
Cyber–physical systems
Threat models
Treaty verification protocols

ABSTRACT

This paper describes a framework for modeling the security of a cyber–physical system in which the behavior of the adversary is controlled by a threat model that captures – in a unified manner – the cyber aspects (with discrete values) and the physical aspects (with continuous values) of the cyber–physical system. In particular, the framework addresses combined (dependent) vector attacks and synchronization/localization issues. The framework identifies the cyber–physical features that must be protected according to the prevailing security policy. Also, the framework can be used for formal proofs of the security of cyber–physical systems.

1. Introduction

The rapid growth of information and communication technologies has prompted the expansion of networked computer systems that address real-world applications, including physical and social applications. This has led to the integration of computing and communication technologies with physical processes under the term "cyber–physical system" (CPS).

CPSs capture novel aspects of networked systems that integrate distributed computing systems with monitoring and controlling entities in the physical environment. For example, in real-time control systems, a hierarchy of sensors, actuators and control processing components are connected to centralized control stations. Other examples include smart grid systems and supervisory control and data acquisition (SCADA) systems that monitor electric power, oil and gas transportation, water distribution and wastewater treatment systems.

Historically, these systems relied on proprietary technologies and were implemented as stand-alone networks in physically protected locations. However, the situation has changed considerably – commodity hardware, software and communication technologies are used to enhance the connectivity of these systems and improve their operation.

Prior work on control systems that focuses on reliability and resilience – by protecting CPSs against random, independent or benign faults and failures of cyber/physical components [8,31] – fails to adequately address integrity, confidentiality and denial-of-service threats [13,14,22,25,34,44]. Moreover, traditional computer and network security approaches do not address in a unified manner how systems can outlive malicious attacks (survivability) and how they can recover from attacks (recoverability) [23,25,34,53].

Securing a CPS goes beyond securing the individual system components. A motivated and highly skilled attacker may use a multi-vector attack that exploits the weaknesses of individual components (e.g., physical and cyber components). Each facet of the attack may not pose a serious threat to the corresponding component; the combined effect, however, may be catastrophic (the attack vectors may be dependent).

An example of a multi-vector attack is Stuxnet [24], which targeted Iran's nuclear centrifuges. In this attack, a worm that used zero-day exploits spread to Windows machines via a local area network or USB drives, carrying a malware payload that infected and reprogrammed programmable logic controllers. Another example is the attack on the SCADA system at a sewage treatment facility in Maroochy Shire,

*Corresponding author. Tel.: +1 850 644 6410; fax: +1 850 644 0058.
E-mail address: burmester@cs.fsu.edu (M. Burmester).

1874-5482/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ijcip.2012.08.002

Queensland, Australia, which caused 80,000 l of raw sewage to be released into local rivers and parks [49]. Yet another example of a multi-vector attack is the SQL Slammer worm, which affected a private computer network at the Davis-Besse nuclear power plant in Oak Harbor, Ohio [39].

There have been many efforts to ensure the security of CPSs. These are primarily based on extending mechanisms that are already used to protect the separate (cyber and physical) components of CPSs. However, there is no formal security model for CPSs that addresses security in a unified framework, and that deals with software threats, hardware threats, network threats and physical threats, possibly combined. Note, however, that several researchers have highlighted the difficulties involved in securing physical systems, in particular with regard to timing attacks [28,38,50], non-interference [26] and execution monitoring [28,36,37].

This paper attempts to address the gap by extending the traditional Byzantine faults model for cryptographic applications to CPSs. In the Byzantine faults paradigm, a CPS is represented as a set of linked abstract machines (graph), some of which may be faulty. The messages exchanged by the CPS components are represented by formal expressions.

The adversary is assumed to be active and has full control of the faulty components. In particular, the adversary can eavesdrop on, intercept and corrupt any message; the only constraints are imposed by the cryptographic methods that are used. The adversary may be computationally unbounded, polynomially bounded or bounded by the inability to solve a particular "hard" problem. To achieve reliable (robust) communications in a system with n components with an adversary who is computationally unbounded, the number of faulty components t should be less than $n/2$ for a fully connected system. The proposed model focuses on the protocol layer and deals with attacks at this layer that result from interactions between an active attacker and the system parties in a possibly unbounded number of parallel sessions. Note that variants of this model exist in which the power of the adversary is restricted (e.g., passive adversary).

A slightly different model was proposed by Herzberg et al. [30]. In their work, the adversary is computationally bounded and the faulty components are periodically repaired (e.g., compromised keys are refreshed). Security is assured if, throughout the lifetime of the system, the adversary is restricted to compromising $t < n/2$ components of the system (different components may be compromised at different times) where n is the number of components. This model captures a physical aspect of the system, especially if the faults are considered to be caused by adversarial operators (insiders) and the components are repaired periodically.

Traditional threat models are restrictive and do not adequately capture the security of CPSs. In particular, they typically exclude survivability and recovery. For example, abnormal behavior may be tolerated by a CPS: a system may transition to critically vulnerable (unsafe) states, but converge to a safe state in the course of time or with some probability. Furthermore, in the Byzantine faults model, the number of faulty components cannot be reduced; in a physical system, however, nodes may become non-faulty in a dynamic way (e.g., after sporadic human intervention or because of Nature).

This paper begins by discussing the inadequacies of traditional adversary models for CPSs. Next, it presents a high level threat model that captures adversarial behavior in a CPS and addresses multi-vector threats to multi-component systems. Finally, it demonstrates how the adversarial threat model can be used to secure a typical CPS.

2. Threat model for cyber–physical systems

A CPS is a finite state system consisting of several networked components, some of which may be cyber while others are physical. A CPS can be modeled by a finite, hybrid timed automaton $\mathcal{A}$ [3,7,29] with faults:

$$\mathcal{A} = (\tau, A, Q, q_0, D, \mathcal{F})$$

where $\tau : t_1, t_2, \ldots$ is a strict monotone unbounded time schedule of positive real numbers; A is a finite set of actions that includes a special symbol $\perp$; $Q \neq \emptyset$ is a finite set of states that is partitioned into safe states Q_s, critical states Q_c and terminal states Q_t; $q_0 \in Q_s$ is an initial state; and $D \subset Q \times Q \times A$ is a transition function that is time triggered, i.e., for $(q, q', a) \in D$ and $t_i \in \tau$:

$$D(t_i) : \; q \xrightarrow{t_i, a} q'$$

describes the transition that action a causes at time t_i.

Critical states are unsafe states from which the system can recover; terminal states are unsafe states from which the system cannot recover. $\mathcal{F}$ is the faults distribution of the CPS, which corresponds to component failure. The transition function D is deterministic when $a \in A \backslash \{\perp\}$ and probabilistic when $a = \perp$. When $a = \perp$, the posteriori state q' is selected by Nature using the distribution $\mathcal{F}$.

A timed execution of $\mathcal{A}$ is a path that starts at time t_1 from state q_0:

$$r : \; q_0 \xrightarrow{t_1, a_1} q_1 \xrightarrow{t_2, a_2} q_2 \xrightarrow{t_3, a_3} q_3 \cdots \longrightarrow q_{i-1} \xrightarrow{t_i, a_i} q_i \cdots$$

and traverses the states q_i instantiated by actions a_i at time t_i.

The parties involved in a CPS are those specified by the system (operators), the adversary (an entity who controls all parties that do not adhere to the system policies/specifications) and Nature (the environment).

We use the game theory paradigm to model Nature: In particular,

- Nature uses the probability distribution $\mathcal{F}$ to randomly select from among her strategies for component failure.
- Nature controls the temporal and location aspects of all events and schedules the state transitions in a timely manner according to the time schedule τ.
- Nature resolves concurrency issues by linking events to their real start time. If two events take place during $(t_{i-1}, t_i]$, then Nature schedules them according to the order in which they occurred. Note that we assume that only a countable number of events are related to the execution of $\mathcal{A}$, so their start time is a sparse subset of the real-time set (positive real numbers).

The threat model for a CPS must capture the system features potentially leading to system failure as well as the adversarial intent. System failure can result from actions by Nature, the adversary or both (the adversary can manipulate

Nature as in a terrorist attack). The adversary can be passive or active. A passive adversary is restricted to eavesdropping on communication channels. An active adversary can additionally modify the contents of the communication channels and use compromised components to undermine the security of the system.

The threat model restricts the adversary to exploiting specific system vulnerabilities. These are identified by:

- System security policies (e.g., availability of services, resilience and privacy of records).
- Vulnerability assessments [19].
- Grey-box penetration testing [19].

The vulnerabilities involve the components of the system, such as control systems, embedded systems and communication channels, but may also involve the system $\mathcal{A}$ as a whole. The security goal for $\mathcal{A}$ is to prevent the adversary from exploiting these features.

Let $V=\{v_1,v_2,\ldots,v_m\}$ be the set of identified features of the states of Q that are vulnerable and must be protected. The features v_i are vectors with discrete and/or continuous values. The vulnerabilities of a CPS may be time-dependent. That is, the adversary may only be able to access $v_i \in V$ at some time t_i. Note that an insider may only be able to access system software while it is being serviced/upgraded.

To identify the vulnerabilities at time t_i we use the function:

$$f:\ (\tau,Q)\mapsto(\tau,2^V);\quad (t_i,q_i)\ \rightarrow f_i(q_i)=V_i\subseteq V$$

which specifies the vulnerabilities of state q_i that the adversary can exploit during the time interval $(t_{i-1},t_i]$.

The threat model for $\mathcal{A}$ is defined by a timed vulnerabilities transition function $D_f(\tau)$:

$$D_f(t_i):\ f_{i-1}(q_{i-1})=V_{i-1}\xrightarrow{t_i,a}f_i(q_i)=V_i$$

which specifies the priori and posteriori features of an adversarial exploit/attack during the time interval $(t_{i-1},t_i]$ (Fig. 1).

In a passive attack, the adversary can eavesdrop on the priori $f_{i-1}(q_{i-1})$-features and the posteriori $f_i(q_i)$-features, and no more. In an active attack, the adversary can also cause a transition $D_f(t_i)$ and exploit the priori and posteriori features. We assume that the adversary has prior knowledge of the vulnerabilities $v_i \in V$ of the system and the structure of $D_f(t_i)$, but not necessarily their values v_i.

Definition 1. An adversary who is restricted to the vulnerabilities of the transitions $D_f(\tau)$ is called a "$D_f(\tau)$-adversary." The automaton $\mathcal{A}$ is $D_f(\tau)$-tolerant if it operates as specified in the presence of a $D_f(\tau)$-adversary.

The specifications for the automaton $\mathcal{A}$ typically require that the system should never enter into a terminal state, and that it should not stay in a critical state for longer than a certain time period. $D_f(\tau)$-tolerance guarantees resilience against adversaries who attempt to exploit the vulnerabilities $v \in V$ of $\mathcal{A}$ and cause it to transition to a state that violates its specifications.

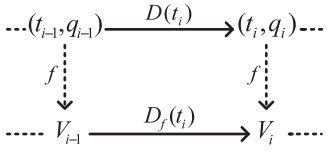


Fig. 1 – Mapping f identifying the priori/posteriori vulnerabilities of the states q_{i-1}, q_i of transition $D_f(\tau)$.

Traditional threat models for cyber systems such as the Byzantine faults model [21] do not capture physical aspects/features/behaviors. For example, the state of a system that uses a wireless medium for communication (e.g., a sensor or RFID system) contains discrete values extracted from continuous values (e.g., RF waveforms). Several attacks can exploit such physical system aspects. One example is an online man-in-the-middle relay attack [6,33], where the adversary interposes himself between the communicating parties and relays messages. Other examples are side channel and power analysis attacks [42], where the adversary exploits information leaked during the physical implementation of a protocol. Both these types of attacks are at the physical layer and are typically excluded from cyber threat models (and their security analyses [11]). To protect against such attacks, physical layer mechanisms (such as temporal and/or location mechanisms and screening) are needed.

To motivate our approach, we show how the transition function $D_f(\tau)$ is used to model the vulnerabilities of cyber and cyber–physical systems.

2.1. *Byzantine faults model*

The Byzantine model [21] assumes a system with n (cyber) components and an adversary that may compromise up to $k<n$ components. In this case, the identified vulnerabilities are $f(t_i,q_i)=V_i$, where $V_i\subseteq V$ is the set of faulty components of q_i, $|V|=k$. The threat transition function is

$$D_f(t_i):\ V_{i-1}\xrightarrow{t_i,a}V_i$$

where $V_{i-1}\subseteq V_i$. That is, an adversary who has compromised the components of V_{i-1} is restricted to attacking states with $V_i\supseteq V_{i-1}$. This defines the allowable system transitions that the adversary can exploit. Note that faulty components cannot recover in this model.

For the model proposed by Herzberg et al. [30], the state of the system is repaired/refreshed at regular intervals. This relabels the faulty components. We then obtain $f(t_i,q_j)=z_i$, $0\le z_i\le k$, which is the number of faulty components.

For this model, the vulnerabilities transition function is

$$D_f(t_i):\ z_{i-1}\xrightarrow{t_i,a}z_i$$

where z_i can be any number in $[0:k]$, if the system has been refreshed during $(t_{i-1},t_i]$. Otherwise, $0\le z_{i-1}\le z_i\le k$. In this threat model, the transitions allow for a reduction in the number of faulty components. For example, if at some point in time the number of faulty components is $z_i\le k$, then in the next time period there may be no faulty component (if faulty components are replaced with non-faulty components). This captures the behavior of certain types of physical faults (e.g., faults that can be fixed).

Such models are typical of physical systems that may tolerate critical state levels provided the system can recover (e.g., the faults are fixed and their duration is short). In this

cyber–physical model, the duration is enforced by Nature and cannot be manipulated by the adversary.

In the Byzantine model, the adversary controls the communication channels of the system: which messages are sent, which messages are received, to whom or by whom, as well as which messages are compromised by faulty components (devices and/or channels). This applies to our model as well when the communication channels are identified as vulnerabilities.

2.2. Threat transitions for network traffic

For this model, $f(t_i,q) = (z_1, \ldots, z_n, Z)$, where z_j is the number of packets sent by node N_j $(j = 1, \ldots, n)$ in a network domain; and Z is the traffic volume in the domain (in packets) during the time interval $(t_{i-1}, t_i]$. We distinguish three cases for z_j:

$$c_1 : z_j \leq a, \quad c_2 : a < z_j \leq b, \quad c_3 : b < z_j$$

with $0 \leq a < b$, where a is an upper bound for normal use and b is the maximum tolerated value for packet transmissions (permitted for short periods only); and three cases for Z:

$$C_1 : Z \leq A, \quad C_2 : A < Z \leq B, \quad C_3 : B < Z$$

with $0 \leq A < B$, where A is a threshold for domain traffic and B is the maximum tolerated level.

States for which the constraint C_3 holds are terminal and lead to domain shutdown. Similarly, nodes that violate the constraint c_3 are denied access to the domain. (For this model, the system can easily recover from a shutdown.) States for which C_2 holds are critical. The thresholds a,A are such that $Z \leq A$ if and only if, for all nodes N_j, we have $z_j \leq a$. The system specifications require that when the state of the system is critical $(A < Z \leq B)$, then all nodes N_j for which $z_j > a$ reduce the number of packets sent to $\leq a$ at time t_i. Finally, states bound by C_1 are safe provided that all the z_j are bounded by the constraints c_1 or c_2. For this model, the vulnerabilities transition function:

$$D_f(t_i) : \ (z_1, \ldots, z_n, Z) \xrightarrow{t_i, a} (z'_1, \ldots, z'_n, Z')$$

requires that the priori and posteriori states are not terminal, and that if a priori state is critical, then the posteriori state must be safe (so $z_j > a$ implies $z'_j \leq a$).

This restricts the adversary to attacking states for which the traffic volume Z is bounded by A over time. $D_f(\tau)$-tolerance is achieved by requiring that, whenever the traffic volume exceeds A, all nodes N_j for which $z_j > a$ reduce the number of packets sent to $z'_j \leq a$ at time t_i.

This model addresses attacks in which the adversary may try to exploit the dependence between the vulnerabilities z_j and Z (e.g., when some nodes send $z_j : b \geq z_j > a$ packets (constraint c_2) and the traffic load is critical (constraint C_2)). This behavior is checked by restricting the adversary to transitions that lead to states with lesser traffic loads.

The network is allowed to stay in a critical state for short periods (one time interval in this case). This is a temporal feature that captures a physical security aspect that is normally excluded from the threat model of cyber systems. Also, this feature highlights one of the main differences between cyber and physical systems.

3. Protecting a natural gas grid

The example described in this section is motivated by the Russia–Ukraine dispute over the price of natural gas and the cost of its transportation. The dispute threatened gas supplies to the European Union (EU) from 2005 to 2010 [17]. Russia provides approximately one-quarter of the natural gas consumed in the EU, and 80% of this is piped across Ukraine to reach the EU. The Russia–Ukraine grid starts in Russia and branches in Ukraine, with one branch going to the EU while the other is for domestic supplies. For its services managing the natural gas grid within its territory, Ukraine is allocated a certain amount of natural gas, which it draws from the pipeline grid.

Fig. 2 shows the EU pipeline with two branches, one for North EU (subnetwork B) and the other for South EU (subnetwork C). Note that a slightly different application was investigated and analyzed in [1].

Subnetwork A supplies natural gas from Russia and subnetwork D provides Ukraine with its allocation. We refer to this as the RU natural gas grid or simply the RU grid.

Let $flow_A$, $flow_B$, $flow_C$ and $flow_D$ be the flows in the subnetworks A, B, C and D, respectively. The Ukraine entitlement $flow_D$ is 10% of the natural gas that flows to North EU and 5% of the natural gas that flows to South EU, i.e.,

$$flow_D = 10\% flow_B + 5\% flow_C.$$

Flow control systems (FCSs) automate and control the flow of natural gas in the pipeline and enforce the flow agreements. A FCS has sensors, actuators, an embedded programmable logic controller (PLC) and a transceiver. The PLC controls the gas flow in its section of pipeline and communicates with neighboring FCSs to regulate the overall flows. It can execute commands that raise or lower the gas flow. Three flow controllers, FCS_A, FCS_B and FCS_C, are controlled by Russia; they regulate the flows coming from Russia and going to North and South EU, respectively. A fourth flow controller FCS_D, controlled by Ukraine, regulates the natural gas allocated to Ukraine. All four controllers are located in Ukraine.

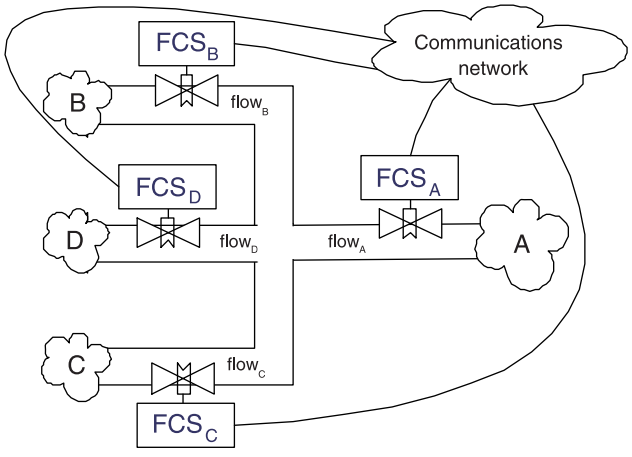


Fig. 2 – The Russia–Ukraine natural gas grid with subnetworks: *A* (Russia), *B* (North EU), *D* (Ukraine) and *C* (South EU).

3.1. Safety specifications (SAF)

- The value of $flow_i$ ($i \in \{A,B,C,D\}$) should not exceed the critical threshold flow level and should normally be within a safe range. The thresholds and ranges are system parameters.
- The condition $0 \leq flow_A - flow_B - flow_C - flow_D < \varepsilon$ must hold, where ε is a small flow variation corresponding to typical gas leakages/fluctuations. The flow variation ε is also a system parameter.

3.2. Security specifications (SEC)

- *Flow privacy*: The values of $flow_A$, $flow_B$ and $flow_C$ should not be inferable from signals transmitted by the flow controllers FCS_A, FCS_B and FCS_C.
- *Flow integrity/verifiability*: At all times, $flow_D - 10\% flow_B - 5\% flow_C < \varepsilon$. Furthermore, Ukraine should be able to verify correctness.

3.3. Threat model ($D_f(\tau)$-adversary)

The vulnerabilities identified by the system specifications of the RU grid concern the flows, $flow_A$, $flow_B$, $flow_C$ and $flow_D$ (SAF), and its communication channels (SEC). In particular,

$$f(state) = (flow_A, flow_B, flow_C, flow_D, z_5, z_6)$$

with

$$z_5 = flow_A - flow_B - flow_C - flow_D$$

and

$$z_6 = 20 flow_D - 2 flow_B - flow_C.$$

The constraints for the safe, critical and terminal flow levels are specified by

$$\begin{aligned}
&c_1 : 0 \leq flow_A < y_1, \quad c_1' : y_1 \leq flow_A < y_1', \quad c_1'' : y_1' \leq flow_A,\\
&c_2 : 0 \leq flow_B < y_2, \quad c_2' : y_2 \leq flow_B < y_2', \quad c_2'' : y_2' \leq flow_B,\\
&c_3 : 0 \leq flow_C < y_3, \quad c_3' : y_3 \leq flow_C < y_3', \quad c_3'' : y_3' \leq flow_C,\\
&c_4 : 0 \leq flow_D < y_4, \quad c_4' : y_4 \leq flow_D < y_4', \quad c_4'' : y_4' \leq flow_D,\\
&c_5 : 0 \leq z_5 < \varepsilon, \quad c_5' : \varepsilon \leq z_5,\\
&c_6 : 0 \leq z_6 < \varepsilon, \quad c_6' : \varepsilon \leq z_6,
\end{aligned}$$

where y_i, y_i' ($i = 1,2,3,4$) are system parameters with:

$$y_i \leq y_i', \quad y_2 + y_3 + y_4 \leq y_1, \quad y_2' + y_3' + y_4' \leq y_1'.$$

States that are bounded by the constraints c_i ($i \in \{1,\ldots,6\}$) are safe. States bound by c_i' ($i \in \{1,\ldots,4\}$) are critical and require an action to reduce flows $flow_A$, $flow_B$, $flow_C$, and $flow_D$ should be reduced proportionately to the levels of the constraints c_i ($i \in \{1,\ldots,5\}$), while maintaining c_6.

Finally, states for which one of the c_1'', c_2'', c_3'', c_4'', c_5', c_6' holds are terminal. When c_1'', c_2'', c_3'' or c_4'' hold, the flow in one of the subnetworks of the pipeline grid exceeds the safety levels. When c_5' holds, the pipeline grid has a leak that exceeds the safety levels. When c_6' holds, the flow of natural gas to Ukraine exceeds the allowed levels based on the contractual allocation. If the system transitions to a terminal state, then it shuts down and all flows are reduced to zero.

The security specifications SEC require that Ukraine not have access to the values of the flows to South EU and North EU. Also, SEC requires Ukraine to be able to verify that it receives its correct allocation of natural gas.

3.4. Verification with privacy

Several cryptographic mechanisms can be used to support an application in which one party (Ukraine) can verify the correctness of a particular value (its gas allocation) without obtaining any additional information about other component values (gas flows to South EU and North EU).

Clearly, Ukraine may obtain such information using covert channels, for example, by accessing the pipelines directly or by accessing accounts/receipts and payments made by South EU and North EU, if these are available.

The threat model does not address security aspects that are not part of the security specifications. Also, it assumes that the RU agreement protocol is based only on readings taken at FCS_B, FCS_C and FCS_D. However, if covert channels are an issue, then the system vulnerabilities must take this feature into account – this extends the scope of a $D_f(\tau)$-adversary and $D_f(\tau)$-tolerance requires additional protection. It is important to note that the challenge of preventing covert channels should not be underestimated, especially in cases where it is possible to collect information leaked from third parties (e.g., through payments made). The issue here is that such information cannot be used to violate the treaty (although it may provide side information). A similar, albeit strategic, issue related to the SALT II Treaty is discussed below.

We now describe a cryptographic protocol that can be used by Ukraine to verify the correctness of flows while providing privacy to Russia. The security of this protocol reduces to the Decision Diffie–Hellman (DDH) assumption.

Definition 2 (*DDH-assumption*). Let G_q be a cyclic group of prime order q with generator g. The "DDH-assumption" concerns the indistinguishability of tuples of type $\langle g, g^x, g^y, g^{xy} \rangle$, $0 \leq x,y < q$, called DH-tuples, from general tuples $\langle g, g^x, g^y, g^z \rangle$, $0 \leq x,y,z < q$. Let D_0 be the set of DH-tuples and D_1 the set of non-DH tuples (with $z \neq xy \bmod q$). A "DDH-distinguisher" $\mathcal{D}$ is a probabilistic polynomial-time algorithm (in the length $|q|$ of q) that upon input of a tuple $T \in D_i$ (i is a random bit) predicts its type with probability better than 1/2. More specifically, there is a constant $\alpha > 0$, such that for sufficiently large q, upon input of a tuple T selected uniformly from $\mathcal{D}_i$, we have

$$\Pr[\mathcal{D}(T) = i \,|\, T \in D_i] > \tfrac{1}{2} + |q|^{-\alpha}, \quad i \in \{0,1\}$$

where $|q|^{-\alpha}$ is a "non-negligible" quantity. The DDH-assumption is that for some families of groups G_q (including the one considered below) there is no DDH-distinguisher (see [9]).

The Flow Verification Protocol defined below uses a family of multiplicative integer groups $Z_p^*(\cdot)$ whose modulus is a "safe" prime p, i.e., $p = 2q+1$ where q is a prime. Let $g \in Z_p$ have order q and G_q be the subgroup generated by g. Let $b = flow_B$, $c = flow_C$, $d = flow_D$, where b and c are rounded to integer values and $2b + c \ll q$.

3.5. Flow verification protocol for RU grid

FCS_B:

- Read flow b.
- Select t_b uniformly from Z_q.

- Compute $y_b = g^{2bt_b}$.
- Send to FCS_C: y_b.

FCS_C:

- Read flow c and message y_b.
- Select s_c, t_c uniformly from Z_q.
- Compute $x_c = g^{s_c}$, $y_c = y_b^{t_c} = g^{2bt_bt_c}$.
- Send to FCS_B: y_c.

FCS_B:

- Read message y_c.
- Compute $z_b = y_c^{t_b^{-1}} = g^{2bt_c}$.
- Send to FCS_C: z_b.

FCS_C:

- Read z_b.
- Compute $z_c = z_b^{t_c^{-1} \cdot s_c} \cdot x_c^c = g^{(2b+c)s_c}$.
- Send to FCS_D: (x_c, z_c).

FCS_D:

- Read flow d and (x_c, z_c).
- Compute $z_d = x_c^{20d}$.
- If $z_d = z_c$ then send to the verifier valid.

This protocol captures correctness because $20d = 2b+c$. It uses a one-way homomorphic function whose security reduces to the DDH-assumption, as we shall show below. Note that the values of the flows must be securely linked to the time and location of their reading and timestamps should be included in all messages.

Definition 3 (*One-way homomorphic function*). A mapping $F : G \to H$ from a group $G(+)$ to a group $H(\cdot)$ is a "one-way homomorphism" if:

(i) F is one-way, i.e., it is infeasible for a probabilistic polynomial-time algorithm to invert any $y = F(x)$, and
(ii) $F(x + y) = F(x) \cdot F(y)$ for all $x, y \in G$.

In the Flow Verification Protocol, the flow controllers FCS_B and FCS_C generate the proof. The verifier (Ukraine) employs FCS_D to verify the proof.

We assume that the flow controllers in the RU grid are tamper-resistant, that FCS_B and FCS_C are managed by Russia and that FCS_D is managed by Ukraine. Note that, although all three FCSs are located in Ukraine, Ukraine has physical access only to FCS_D while Russia has access to FCS_A, FCS_B and FCS_C. We also assume that the embedded PLCs are trusted and autonomous. Moreover, the FCS components can be checked by all the concerned parties prior to deployment. In addition, the communication channels between the FCSs are reliable and authenticated; this can be achieved by employing redundancy and using cryptographic authentication mechanisms such as message authentication codes or digital signatures, but digital signatures must be used for validation. Finally, the communications can be over fixed lines or wireless.

Interestingly, the RU grid situation has some commonalities with diplomacy during the Cold War. The Strategic Arms Limitation Treaty (SALT II) between the United States and the Soviet Union (1977–1979) sought to curtail the number of intercontinental ballistic missiles (ICBMs) to 2250 on each side. This involved installing tamper-resistant sensor control units in the ICBM silos to detect the presence of missiles. The sensors were to be used to verify the number of deployed ICBMs. Both parties would have had access to this information, but to no other information, especially regarding the locations of the responding silos [10,20,48], prior to deployment.

The $D_f(\tau)$-adversary can be any party other than the prover. In our scenario, the adversary is an insider (possibly Ukraine) who knows the value of $2b+c$ (this should be $20d$, where d is the amount of natural gas allocated to Ukraine). The goal of the adversary is to undermine the privacy of the flows b and c.

Theorem 1. *Suppose that:*

(i) *FCS_A, FCS_B, FCS_C and FCS_D are tamper-resistant, and the*
(ii) *Communication channels linking FCS_A, FCS_B, FCS_C and FCS_D are reliable and authenticated.*

Then, the RU pipeline grid will tolerate an $D_f(\tau)$-adversary.

Proof. The first requirement implies that the adversary cannot access the inner state of the FCS (e.g., the values of b and c or the randomness t_b, s_c, t_c used to compute their outputs). The second requirement is that transmissions are reliable and the origins of messages can be established.

The embedded programmable logic controllers of the FCS can be designed to enforce $D_f(\tau)$-tolerance since we assume that: (i) they are not faulty; (ii) their communication channels are trusted; and (iii) the system is autonomous. Insider threats to the FCS are thwarted because the system is autonomous with tamper-proof components. □

Theorem 2. *Suppose the RU pipeline grid is $D_f(\tau)$-tolerant and:*

(i) *no covert channels leak the values b and c, and the*
(ii) *DDH-assumption holds.*

Then, the Flow Verification Protocol is correct and provides privacy for the flows b and c against an eavesdropping $D_f(\tau)$-adversary who knows the value of the flow d.

Proof. The first assumption states that the $D_f(\tau)$-adversary cannot find the values of b and c by using some other means, external to the protocol, e.g., by accessing the pipelines directly or monitoring the EU gas consumption/payments.

Correctness follows from the fact that $20d = 2b+c$.

Regarding the privacy of b and c, suppose that an eavesdropping $D_f(\tau)$-adversary $\mathcal{E}$ can access the values:

g^{2bt_b}, g^{2bt_c}, $g^{2bt_bt_c}$ and g^{s_c}, $g^{(2b+c)s_c}$

of the communication channels of the RU-grid.

Since we assume that $\mathcal{E}$ knows the value of d and $20d = 2b+c$, the last two values do not contribute any

additional knowledge regarding the values of b and c. To prove the privacy of b in the presence of $\mathcal{E}$, we consider an experiment $\mathsf{Priv}^{\mathsf{eav}}_{\mathcal{E}}$ in which $\mathcal{E}$ chooses two values b_0, b_1 of b, and is then given the obfuscated tuple:

$$T_{b_i} = \langle g, g^{2b_i t_b}, g^{2b_i t_c}, g^{2b_i t_b t_c} \rangle$$

of one of these values, where i is a random uniform bit. In this experiment, the adversary $\mathcal{E}$ (using a probabilistic polynomial-time algorithm) must find which one of b_0, b_1 was used in T_{b_i}. Note that the indistinguishability of obfuscated data by a polynomial-time adversary captures a strong aspect of privacy and is the basis for semantic security.

Of course, $\mathcal{E}$ can toss a coin to guess which value was encrypted. He would succeed with probability 1/2. Suppose that $\mathcal{E}$ can find the correct value b_i with probability $1/2 + \varepsilon$, $\varepsilon = \varepsilon(|q|)$. We show that ε is negligible (in $|q|$) by reducing $\mathcal{E}$ to a DDH-distinguisher $\mathcal{D}$.

Let $T = \langle g, g^x, g^y, g^z \rangle$ be the G_q-tuple input to the distinguisher. $\mathcal{D}$ must decide if this is a DH-tuple (i.e., $z = xy \bmod q$) or not. For this purpose, $\mathcal{D}$ queries $\mathcal{E}$ for two values b_0, b_1 and then computes the tuple:

$$T'_{b_i} = \langle g, g^{2b_i x}, g^{2b_i y}, g^{2b_i z} \rangle$$

where i is a random bit. The distinguisher $\mathcal{D}$ gives T'_{b_i} to the adversary $\mathcal{E}$ in the experiment $\mathsf{Priv}^{\mathsf{eav}}_{\mathcal{E}}$ instead of T_{b_i}. If $\mathcal{E}$ predicts that the value b_i was used in the computation of T'_{b_i}, then $\mathcal{D}$'s prediction is that T is a DH-tuple ($z = xy \bmod q$). $\mathcal{D}$ outputs 1. Otherwise, $\mathcal{D}$ tosses a coin and bases its prediction on the outcome of the toss (0 or 1). It is easy to see that the probability that $\mathcal{D}$ can distinguish DH-tuples (output 1) is $1/2 + \varepsilon/2$ since $\mathcal{E}$ succeeds with probability ε whenever T is a DH-tuple. Then, by the DDH-assumption, $\varepsilon/2$ and, hence, ε must be negligible (in $|q|$). This completes the proof. □

Remark 1. The Flow Verification Protocol is a proof that the (cyber) equation: $20d - 2b - c = 0$ holds, whereas for correctness it is necessary to show that the (physical) inequality: $0 \leq 20d - 2b - c < \varepsilon$ holds. A simple fix exists for this particular application. However, in general, using a cyber mechanism (cryptography) to secure a physical system may be inadequate, and we may have to use hybrid security mechanisms.

To show that the proof is valid, we first sandbox the Flow Verification Protocol to separate it from the $D_f(\tau)$-tolerance supporting mechanisms. We then calculate the flows using a unit of measurement for which $1/4$ unit $> \varepsilon$. We take integer values and map these to Z_q. For example, if a flow measurement is x, it is first reduced by using a new measurement unit to obtain x' units, and then it is reduced to its integer value $x_v = \lceil x' \rceil$ in Z_q. This approach is good enough for applications in which the fluctuations in measured values are small. Observe that the exact flow values x as measured at the FCSs are used to prove $D_f(\tau)$-tolerance.

Remark 2. A change of flow in the flow controller FCS_A will only register at one of the flow controllers FCS_B, FCS_C or FCS_D at a later time [43]. To deal with time dependencies of flows, the values of $flow_A$, $flow_B$, $flow_C$ and $flow_D$ are time-stamped, and these delays should be taken into account when verifying the values of the flow allocations.

4. Related work

In a CPS, distributed computing components interact with the physical environment. Several approaches have been proposed for modeling a CPS. These are described below.

A hybrid automaton [2,29,40] is a formal model that combines finite state transition systems with discrete variables (whose values capture the state of the modeled discrete or cyber components) and continuous variables (whose values capture the state of the modeled continuous or physical components).

Another related formalism, timed automata [3,32], can be used to model the timing properties of CPSs. Such machines are finite automata with a finite set of real-valued clocks. They accept timed words, i.e., infinite sequences in which a real-time of occurrence is associated with each symbol.

Hybrid process algebras [5,18] are a powerful tool for reasoning about physical systems and provide techniques for analyzing and verifying security protocols for hybrid automata.

Bond graphs [47] are used to synthesize mixed component systems, with electrical, mechanical, hydraulic, thermal and, more generally, physical components. Bond graphs are domain independent and allow the free composition and efficient analysis and classification of models, permitting rapid determination of various types of feasibility and acceptability of candidate designs.

Genetic programming [35] is an evolutionary algorithm based methodology inspired by biology. It is a powerful tool for creating computer programs that perform specific user-defined tasks. When combined with bond graphs, genetic programming provides for better synthesis of complex mixed component systems.

Hybrid bond graphs [45] combine bond graphs with hybrid automata to provide a uniform, physics-based formal model that incorporates controlled and autonomous mode changes as idealized switching functions.

Security and survivability goals, threats and attacks on CPS control systems, as well as proactive/reactive mechanisms for robust distributed control and distributed consensus in the presence of deception and DoS adversaries, are summarized in [13,14]. A survey of vulnerabilities, failures and attacks on real-time distributed control systems, and mitigation and recovery strategies is given in [34]. A taxonomy of attacks against energy control systems is presented in [25]. Data replay threats on control systems are studied and formulated in [44]. A comprehensive, albeit informal, threat model and a taxonomy of attacks against sensor networks in SCADA systems are described in [15]. Monitoring and intrusion/anomaly detection methodologies and automatic response for control systems and a formalism for anomaly detection is given in [13]. In [13], risk assessment formalisms are also proposed for measuring the possible damage caused by cyber attacks on control systems.

In [31], failures and fault tolerance in distributed CPSs are modeled, where a CPS is modeled as a distributed algorithm executed by a set of agents and the continuous dynamics of the CPS is abstracted as discrete transitions. An informal attack model for energy control systems is given in [25],

where attacks are related to the vulnerabilities they exploit and the damage they cause. Finite state machine models based on Petri nets have also been proposed to describe cyber attacks [52]. Other attack models include attack trees [46], where the root node denotes the goal of an attacker and a path from leaf nodes to the root node denotes an attack instance, i.e., the steps for completing the attack [51]; a critique of attack trees is presented in [12]. A model using graph theory to express control system failures and attacks is also presented in [12]. In [16], a language for modeling multistep attack scenarios on process control systems is proposed, enabling correlation engines to use the models to recognize attack scenarios.

Stochastic approaches have been used to model the probabilities of failures in distributed computing systems [4]. Game theoretic techniques and formalisms for modeling attacks and defense strategies in CPSs are described in [41]. Here, the game is between an attacker and the defender of a CPS, where the attacker tries to disrupt either the cyber or physical components.

Finally, access control and information flow based policies for CPS security are analyzed in [1,27], and a framework for enforcing information flow policies in CPSs in order to obfuscate the observable effects of a system is presented in [27].

5. Conclusions

The threat framework proposed for CPSs is based on the traditional Byzantine paradigm for cryptographic security in which the basic security features and requirements as specified by the security policies are used to identify system vulnerabilities. An important benefit of the framework is that it supports formal analyses and security proofs using existing cryptographic methodologies.

Acknowledgments

This research was partially supported by the National Science Foundation under Grant no. DUE 1027217.

REFERENCES

[1] R. Akella, H. Tang, B. McMillin, Analysis of information flow security in cyber–physical systems, International Journal of Critical Infrastructure Protection 3 (3–4) (2010) 157–173.

[2] R. Alur, C. Courcoubetis, T. Henzinger, P. Ho, Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems, in: R. Grossman, A. Nerode, A. Ravn, H. Rischel (Eds.), Hybrid Systems, Springer, Berlin, Heidelberg, Germany, 1993, pp. 209–229.

[3] R. Alur, D. Dill, A theory of timed automata, Theoretical Computer Science 126 (2) (1994) 183–235.

[4] O. Babaoglu, On the reliability of consensus-based fault-tolerant distributed computing systems, ACM Transactions on Computer Systems 5 (4) (1987) 394–416.

[5] J. Baeten, B. van Beek, P. Cuijpers, M. Reniers, J. Rooda, R. Schiffelers, R. Theunissen, Model-based engineering of embedded systems using the hybrid process algebra Chi, Electronic Notes in Theoretical Computer Science 209 (2008) 21–53.

[6] S. Bengio, G. Brassard, Y. Desmedt, C. Goutier, J. Quisquater, Secure implementations of identification systems, Journal of Cryptology 4 (3) (1991) 175–183.

[7] J. Bengtsson, W. Yi, Timed automata: semantics, algorithms and tools, in: J. Desel, W. Reisig, G. Rozenberg (Eds.), Lectures on Concurrency and Petri Nets, Lecture Notes in Computer Science, vol. 3098), Springer, Berlin, Heidelberg, Germany, 2003, pp. 87–124.

[8] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, Diagnosis and Fault-Tolerant Control, Springer, Berlin, Heidelberg, Germany, 2006.

[9] D. Boneh, The decision Diffie-Hellman problem, in: Proceedings of the Third International Symposium on Algorithmic Number Theory, 1998, pp. 48–63.

[10] M. Burmester, Y. Desmedt, T. Itoh, K. Sakurai, H. Shizuya, M. Yung, A progress report on subliminal-free channels, in: Proceedings of the First International Workshop on Information Hiding, 1996, pp. 157–168.

[11] M. Burmester, T. Le, B. Medeiros, G. Tsudik, Universally composable RFID identification and authentication protocols, ACM Transactions on Information and System Security 12 (4) (2009) 1–33.

[12] J. Butts, M. Rice, S. Shenoi, Modeling control system failures and attacks—The Waterloo campaign to oil pipelines, in: T. Moore, S. Shenoi (Eds.), Critical Infrastructure Protection IV, Springer, Heidelberg, Germany, 2010, pp. 43–62.

[13] A. Cardenas, S. Amin, Z. Lin, Y. Huang, C. Huang, S. Sastry, Attacks against process control systems: risk assessment, detection and response, in: Proceedings of the Sixth ACM Symposium on Information, Computer and Communications Security, 2011, pp. 355–366.

[14] A. Cardenas, S. Amin, S. Sastry, Secure control: towards survivable cyber–physical systems, in: Proceedings of the Twenty-Eighth IEEE International Conference on Distributed Computing Systems Workshops, 2008, pp. 495–500.

[15] A. Cardenas, T. Roosta, S. Sastry, Rethinking security properties, threat models and the design space in sensor networks: a case study in SCADA systems, Ad Hoc Networks 7 (8) (2009) 1434–1447.

[16] S. Cheung, U. Lindqvist, M. Fong, Modeling multistep cyber attacks for scenario recognition, in: Proceedings of the Third DARPA Information Survivability Conference and Exposition, 2003, pp. 284–292.

[17] E. Chow, J. Elkind, Where East meets West: European gas and Ukrainian reality, Washington Quarterly (Center for Strategic and International Studies) 32 (1) (2009) 77–92.

[18] P. Cuijpers, J. Broenink, P. Mosterman, Constitutive hybrid processes: a process-algebraic semantics for hybrid bond graphs, Simulation 84 (7) (2008) 339–358.

[19] Department of Homeland Security and Centre for the Protection of the National Infrastructure, Cyber Security Assessments of Industrial Control Systems, Washington, DC 〈www.us-cert.gov/control_systems/pdf/Cyber_Security_Assessments_of_Industrial_Control_Systems.pdf〉, 2010.

[20] W. Diffie, The national security establishment and the development of public-key cryptography, Designs, Codes and Cryptography 7 (2) (1995) 9–11.

[21] D. Dolev, The Byzantine generals strike again, Journal of Algorithms 3 (1) (1982) 14–30.

[22] J. Eisenhauer, P. Donnelly, M. Ellis, M. O'Brien, Roadmap to Secure Control Systems in the Energy Sector, Energetics, Columbia, MD, 2006.

[23] R. Ellison, D. Fisher, R. Linger, H. Lipson, T. Longstaff, N. Mead, Survivable Network Systems: An Emerging Discipline, Technical Report CMU/SEI-97-TR-013, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1997.

[24] N. Falliere, L. Murchu, E. Chien, W32. Stuxnet Dossier, Symantec, Mountain View, California, 2011.

[25] T. Fleury, H. Khurana, V. Welch, Towards a taxonomy of attacks against energy control systems, in: M. Papa, S. Shenoi (Eds.), Critical Infrastructure Protection II, Boston, Massachusetts, 2008, pp. 71–85.
[26] T. Gamage and B. McMillin, Enforcing information flow properties using compensating events, in: Proceedings of the Forty-Second Hawaii International Conference on System Sciences, 2009.
[27] T. Gamage, B. McMillin, T. Roth, Enforcing information flow security properties in cyber–physical systems: a generalized framework based on compensation, in: Proceedings of the Thirty-Fourth Annual IEEE Computer Software and Applications Conference, 2010, pp. 158–163.
[28] K. Hamlen, G. Morrisett, F. Schneider, Computability classes for enforcement mechanisms, ACM Transactions on Programming Languages and Systems 28 (1) (2006) 175–205.
[29] T. Henzinger, The theory of hybrid automata, in: Proceedings of the Eleventh Annual IEEE Symposium on Logic in Computer Science, 1996, pp. 278–292.
[30] A. Herzberg, S. Jarecki, H. Krawczyk, M. Yung, Proactive secret sharing or: How to cope with perpetual leakage, in: Proceedings of the Fifteenth Annual International Cryptology Conference on Advances in Cryptology, 1995, pp. 339–352.
[31] T. Johnson, Fault-Tolerant Distributed Cyber–Physical Systems: Two Case Studies, M.S. Thesis, Department of Electrical and Computer Engineering, University of Illinois, Urbana, IL, 2010.
[32] D. Kaynor, N. Lynch, R. Segala, F. Vaandrager, The theory of timed I/O automata, in: Synthesis Lectures on Distributed Computing Theory, vol. 1(1), 2010, p. 1–137.
[33] C. Kim, G. Avoine, F. Koeune, F. Standaert, O. Pereira, The Swiss-knife RFID distance bounding protocol, in: Proceedings of the Eleventh International Conference on Information Security and Cryptology, 2008, pp. 98–115.
[34] R. Kisner, W. Manges, T. Mcintyre, J. Nutaro, J. Munro, P. Ewing, M. Howlader, P. Kuruganti, M. Olama, Cybersecurity Through Real-Time Distributed Control Systems, Technical Report ORNL/TM-2010/30, Oak Ridge National Laboratory, Oak Ridge, TN, 2010.
[35] J. Koza, Genetic Programming: on the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, 1992.
[36] L. Lamport, Proving the correctness of multiprocess programs, IEEE Transactions on Software Engineering 3 (2) (1997) 125–143.
[37] L. Lamport, Proving possibility properties, Theoretical Computer Science 206 (1–2) (1998) 341–352.
[38] L. Lamport, Real-time model checking is really simple, in: Proceedings of the Thirteenth Advanced Research Working Conference on Correct Hardware Design and Verification Methods, 2005, pp. 162–175.
[39] E. Levy, Crossover: online pests plaguing the offline world, IEEE Security and Privacy 1 (6) (2003) 71–73.
[40] N. Lynch, R. Segala, F. Vaandrager and H. Weinberg, Hybrid I/O automata, in: Proceedings of the DIMACS Workshop on Verification and Control of Hybrid Systems, 1995, pp. 496–510.
[41] C. Ma, N. Rao and D. Yau, A game theoretic study of attack and defense in cyber–physical systems, in: Proceedings of the First IEEE International Workshop on Cyber–Physical Networking Systems, 2011, pp. 708–713.
[42] S. Mangard, E. Oswald, T. Popp, Power Analysis Attacks—Revealing the Secrets of Smart Cards, Springer, New York, 2007.
[43] B. McMillin, Personal communication, 2012.
[44] Y. Mo, B. Sinopoli, Secure control against replay attacks, in: Proceedings of the Forty-Seventh Annual Allerton Conference on Communication, Control and Computing, 2009, pp. 911–918.
[45] I. Roychoudhury, M. Daigle, P. Mosterman, G. Biswas, X. Koutsoukos, A method for the efficient simulation of hybrid bond graphs, in: Proceedings of the International Conference on Bond Graph Modeling, 2007, pp. 177–184.
[46] B. Schneier, Attack trees, Dr. Dobb's Journal 24 (12) (1999) 21–29.
[47] K. Seo, Z. Fan, J. Hu, E. Goodman, R. Rosenberg, Toward an automated design method for multi-domain dynamic systems using bond graph and genetic programming, Mechatronics 13 (8–9) (2003) 851–885.
[48] G. Simmons, Personal communication, 1993.
[49] J. Slay, M. Miller, Lessons learned from the Maroochy water breach, in: E. Goetz, S. Shenoi (Eds.), Critical Infrastructure Protection, Boston, Massachusetts, 2007, pp. 73–82.
[50] H. Tang, B. McMillin, Security property violation in CPS through timing, in: Proceedings of the Twenty-Eighth International Conference on Distributed Computing Systems, 2008, pp. 519–524.
[51] C. Ten, C. Liu, G. Manimaran, Vulnerability assessment of cybersecurity for SCADA systems, IEEE Transactions on Power Systems 23 (4) (2008) 1836–1846.
[52] R. Wu, W. Li, H. Huang, An attack modeling based on hierarchical colored Petri nets, in: Proceedings of the First International Conference on Computer and Electrical Engineering, 2008, pp. 918–921.
[53] K. Xiao, S. Ren, K. Kwiat, Retrofitting cyber physical systems for survivability through external coordination, in: Proceedings of the Forty-First Hawaii International Conference on Systems Science, 2008, p. 465.