2010 34th Annual IEEE Computer Software and Applications Conference Workshops

Enforcing Information Flow Security Properties in Cyber-Physical Systems: A Generalized Framework Based on Compensation

Thoshitha T. Gamage Bruce M. McMillin Thomas P. Roth
{ttgdp5@mail.mst.edu, ff@mst.edu, tprfh7@mst.edu}
Department of Computer Science
Intelligent Systems Center
Missouri University of Science and Technology, Rolla, MO 65409-0350

***Abstract*—This paper presents a general theory of event compensation as an information flow security enforcement mechanism for Cyber-Physical Systems (CPSs). The fundamental research problem being investigated is that externally observable events in modern CPSs have the propensity to divulge sensitive settings to adversaries, resulting in a confidentiality violation. This is a less studied yet emerging concern in modern system security. A viable method to mitigate such violations is to use information flow security based enforcement mechanisms since access control based security models cannot impose restrictions on information propagation. Further, the disjoint nature of security analysis is not appropriate for systems with highly integrated physical and cyber infrastructures. The proposed compensation based security framework is foundational work that unifies cyber and physical aspects of security through the shared semantics of information flow. A DC circuit example is presented to demonstrate this concept.**

I. Introduction

Preserving the confidentiality of sensitive actions/events is a vital aspect of modern system security. By uncovering sensitive actions, adverse parties can identify the crucial components of a system and target specific attacks. Numerous integration levels within modern systems can be broadly categorized into two domains: a cyber infrastructure (computations, control algorithms, decision engines, databases, etc.) and a physical infrastructure (physical processes and components, links and connections, etc.). Commonly known as Cyber-Physical Systems (CPSs), such systems are put together to provide better resource utilization, control, fault tolerance and performance [1].

The proper functionality of CPSs has direct impact on the nations' economic and social stability. In August 2003, an estimated 50 million people in the Midwest and Northeast portions of the United States and Ontario, Canada were affected by a power outage lasting up to 4 days [2]. Although this event was triggered by a cascading failure in power lines loosely coupled with a cyber failure, rather than an intentional act of sabotage, it highlights and prompts the need for extensive research, methodologies, new tools and models of security to safeguard these critical infrastructures.

One prominent feature in CPSs is that embedded computers and communication networks govern both physical manifestations and computations. This, in turn, affects how the two primary infrastructures interact with each other, and the outside world. From a functionality point of view, a CPS can be regarded as the intersection of properly built individual (cyber, physical, network) components. This unfortunately, is not the case in terms of security. Treating CPS security in a disjoint manner, allows unintended information flow.

Access control based methods and information flow based methods are the two primary approaches to system security policies and mechanisms. From a confidentiality standpoint, the problem in CPSs is that certain aspects of the physical portion are always observable. As a side effect, explicit information flow violations take place as externally observable physical manifestations divulge sensitive system settings [3]. Such derived knowledge coupled with the system semantics can be used against the system as integrity and availability attacks. Unfortunately, access control-based security models fail to prevent propagation [4]; they cannot control how the data will be used after been read. A viable alternative is to use flow-based security models such as information flow security enforcement mechanisms.

Even within the information flow enforcement domain, only a handful (if not none) of work is done towards CPS security. CPS security needs to be considered at the system level [5]. Earlier developments in enforceable security properties were strictly based on safety properties [6]. The safety property requirement is too strong for CPS security analysis and prevents Information Flow Security Properties (IFPs) from being enforced and monitored [7]. However, unwinding theorem-based security automata can be designed to capture possibilistic future execution sequences by not having to detect violations at present, but eventually sometime in future [8]. This allows IFPs to be enforced and monitored outside the Alpern-Schneider framework [9].

This paper presents an event compensation based gen-

This work was supported in part by the Future Renewable Electric Energy Distribution Management Center; a National Science Foundation supported Engineering Research Center, under grant NSF EEC-0812121 and NSF CSR award CCF-0614633, and in part by the Missouri S&T Intelligent Systems Center.

978-0-7695-4105-1/10 $26.00 © 2010 IEEE
DOI 10.1109/COMPSACW.2010.36

eralized framework to enforce IFPs in CPSs. The idea is to compensate the observable effect of a (potentially information flow violating) system event by pairing it up with an appropriate reaction event(s). By executing compensating events in a timely (i.e. within a certain time period) and coordinated manner, the expected net observable change is either insignificant in terms of deducing sensitive information or equivalent to some other system characteristic(s). Thus, the objective is to obfuscate the observable effects of a system*.

In this respect, the contributions of this work are,

- Introduce a class of system properties called $\mathcal{P}$–compensate properties which are execution monitoring enforceable in cyber-physical systems
- Develop a semantic model to analyze confidentiality violation in cyber-physical systems
- Extend previous work on runtime enforceable policies by combining a predicate mechanism with the ability to inject events
- Extend existing enforcement mechanisms beyond the safety property requirements by proposing an event compensation based security framework

The work in this paper extends previous work in [3] by presenting a precise system characteristic, the *$\mathcal{P}$–Compensate Property*, required for compensation. The observability analysis in [10] is also extended beyond series and parallel connections to a "mix network". Further, the applicability of the compensation concept to protect Nondeducibility security in CPSs is presented.

Section II lists some of the recent work in enforcing information flow security policies. Section III introduces the proposed framework. The applicability of the proposed work on a CPS is presented in Section IV. Section V presents the correlation between the inherent external observability and the resulting deducibility in CPSs and Section VI shows how compensation can protect Nondeducibility security in a CPS. Section VII lists the conclusion of this work.

II. Related Work

Information flows between processes which are not supposed to communicate should be prevented in multilevel security systems. In other words, the ability to deduce sensitive high-level domain ($\mathcal{D}_H$) information at a low-level domain ($\mathcal{D}_L$) is an information flow security violation. The sensitivity of information and the "communication" depend on each processes' view of the system. The term "deduce" is a generalization for different ways information flow security can be compromised. Fundamental IFPs introduced over the years such as Noninterference, Noninference and Nondeducibility [11] attempt to characterize and capture these "possibilistic" flows.

Information flow property enforcement is two fold: static compile time enforcement and runtime enforcement. Compile time enforcers such as secure-type systems, mechanisms based on petri nets, process algebra and program logic, etc., tend to be too imprecise; a static enforcer can potentially reject a program based on a partial analysis [12].

Conceptually, runtime enforcement monitors work by monitoring the computational steps of untrusted programs and intervening whenever execution is about to violate the security policy being enforced [13]. The earliest threshold on Execution Monitoring (EM) enforceable security policies was established in [6] by stating that only safety properties can be enforced using a monitoring mechanism. A büchi-like *security automata* enforced security policies by terminating the target execution upon detecting a violation. Flow-based security properties are not safety properties [7, 14] and the safety requirement, unfortunately, precludes these from being enforced. For these properties, the decision to terminate an execution can not be purely based on a detected violation on a single execution [7]. Extending the security automata [8, 11] implemented with monitors [15] can enforce non-safety properties.

III. The Framework

The proposed work of this article improves EM security automata [6] by combining it with an emulator [8] and the event insertion capability of the edit automata [15]. Unlike the traditional "conservative" approach†, the framework proposed here employs a more optimistic event compensation based enforcement mechanism $\mathcal{E}_S$.

Event compensation is only applicable to executions which are *eligible for cleansing*. Cleansing an execution allows it to extend beyond a violation point and prevent it from being discarded. An edit automata [15], which can modify the behavior of an execution during runtime (with suppression and injection), is a good example of execution cleansing.

Only a certain class of qualified executions can be extended in this manner. Suppose there exists an execution σ with a distinctly identifiable violation point σ_j, a valid prefix $\sigma[\ldots i]$ and a projected postfix $\sigma[k\ldots]$. The optimistic assumption is that, in the absence of σ_j, the execution maintains property $\mathcal{P}$. This is similar in concept to the suppression operation introduced in [15]. Formally, this characteristic can be denoted as,

$$\sigma = \sigma[\ldots i]\ \sigma_j\ \sigma[k\ldots] \implies \sigma[\ldots i]\ \sigma[k\ldots] \in (\mathcal{P} \in \mathcal{P}_S^*) \tag{1}$$

Executions similar in form to (1) are eligible for cleansing under the proposed enforcement scheme. Mathematically, this allows correction action(s) to be injected immediately after σ_j and extend the execution. However, in order to maintain the desired $\mathcal{P}$, such an injection needs to compensate for the effects of σ_j. By performing event compensation, $\mathcal{E}_S$ restores the system back to an operational state. Thus, such an optimistic view of the system is also a liveness [9] feature.

A. $\mathcal{P}$–Compensate Property

This work quantifies an execution step as a finite sequence of controlled state transitions. By doing so, $\mathcal{E}_S$ is empowered

*This is not to be confused with obfuscating actual physical actions which no amount of compensation can reverse

†The security automata in [6] for example takes a conservative approach by immediately terminating executions

to inject more than one correction action, depending on the requirement and the specific property expected to maintain.

Definition 1. *[**Compensation Sequence**] For some identifiable execution violation point $\sigma_j \in \sigma$, a compensation sequence $\varsigma \in \Sigma^*$ is defined as a finite sequence of states starting with σ_j which can compensate for σ_j for some property $\mathcal{P}$.*

$$\sigma[\dots i]\ \sigma[k\dots] = \sigma[\dots i] \varsigma\ \sigma[k\dots] \quad where,\ \varsigma \in \Sigma^*, \varsigma = \sigma[j\dots]$$

Consequently, associated with ς is a finite sequence of compensating actions $\varphi = < \phi_j^c, \phi_{j+1}^c, \dots, \phi_k^c > \in \Phi^*$ corresponding to each state transition in ς. Thus, a compensated execution takes the following form.

$$\sigma[\dots i] \xrightarrow{\phi_j^c} \sigma_j \xrightarrow{\phi_{j+1}^c} \sigma_{j+1} \dots \xrightarrow{\phi_k} \sigma[k\dots] \quad where,\ \varsigma \in \Sigma^*, \varsigma = \sigma[j\dots]$$

Once an eligible execution is identified, a compensation sequence is calculated to compensate for σ_j. This is done by carefully calculating a φ which can lead the overall system back to a safe state with respect to the IFP $\mathcal{P}$. The compensativeness of a system characterizes the ability to cleanse executions respect to some $\mathcal{P}$.

[3] formally showed the potential of event compensation to preserve IFPs in a CPS. Even though there is a momentarily lapse in the corresponding feature, the compensated execution as a whole would still adhere to the desired $\mathcal{P}$. The idea is that the system as a whole, not individual operations, need to satisfy the required property.

Definition 2. *[$\mathcal{P}$–**Compensate Property**] A system is compensative with respect to a property $\mathcal{P} \in \mathcal{P}_S$ if and only if, for some execution σ with an identifiable violation point at j, there exists a compensation sequence ς.*

$$\exists \sigma \in \Sigma^*,\ \sigma \notin \mathcal{P} : \neg\hat{\wp}(\sigma[\dots j]), \quad \exists \varsigma \in \Sigma^* : \varsigma = \sigma[j\dots],\ \sigma[\dots i]\ \varsigma\ \sigma[k\dots] \in \mathcal{P} \quad (2)$$

B. Compensating Couple [3]

In the most basic form, ς consists of a single element and two associated actions, i.e., $\mid \varphi \mid = 2$. This is formally defined as a compensating couple. With this, the security automata [6] can be extended to a *compensation automata* as follows. The state space $\mathcal{Q}$ is divided into two sets. $\mathcal{W} \subseteq \mathcal{Q}$ is the set of information flow safe states and $\mathcal{U} \subseteq \mathcal{Q}$ is the set of vulnerable(unsafe) states.

Definition 3 (Compensation Automata). *The compensation automata consists of 6-tuples $(\mathcal{Q}, \mathcal{Q}_0, I, \delta, \mathcal{W}, \mathcal{D})$ where,*

- *$\mathcal{Q}$ is a set automaton states*
- *$\mathcal{Q}_0$ is a set of initial states for the automaton $\mathcal{Q}_0 \subseteq \mathcal{Q}$*
- *$I \subseteq \Phi$ is a set of input symbols of the form (ϕ_{i-1}^c, ϕ_i^c) : $(\phi_{i-1}^c, \phi_i^c \in I)$*
- *δ is the a state transition function $\delta : \mathcal{Q} \times I \rightarrow 2^{\mathcal{Q}}$ specified under a predicate $\hat{\wp}()$*
- *$\mathcal{W}$ is a set of final states $\mathcal{W} \subseteq \mathcal{Q}$*
- *$\mathcal{D}$ is the set of security domains*

With respect to a IFP $\mathcal{P}$, the effect of executing a compensating couple needs to be null, i.e., $\phi_i^c - \phi_{i-1}^c = \langle\rangle$. The second event of the pair, ϕ_i^c, needs to lead the state machine back to an information flow secure state as well as compensate for the first event ϕ_{i-1}^c. The predicate $\hat{\wp}()$ is used to make sure that each compensating transition adheres to the particular property $\mathcal{P}$ in concern. This enables the compensation automata to maintain $\mathcal{P}$–compensating property during each "compensating" step of the execution. Thus,

$$\forall j, \neg\hat{\wp}(\sigma[\dots i]\sigma_j[k\dots]) \implies \hat{\wp}(\sigma[\dots i] \xrightarrow{\varphi} \sigma_k) \quad (3)$$

As a side effect of the characteristic equation 2, φ steps through a sequence of insecure states; certain intermediate states of every ς can momentarily violate $\mathcal{P}$. However, the argument is that ς is finite by definition and φ is executed in a timely and controlled manner to avoid external observations. Thus, the effect of the inherent security vulnerability is temporary. Further, each event in a compensating sequence needs to be from the secure domain, i.e., $\forall \phi_i^c \in \varphi, \phi_i^c \trianglerighteq \mathcal{D}_H$. A detailed technical specification of compensation automata is found at [3].

IV. Application : Smart Grid

The emerging "Smart Grid" improves the power grid reliability using a Flexible AC Transmission System (FACTS) network. FACTS devices are reconfigurable/reprogrammable power electronic devices which can change specific transmission line parameters [16]. Each FACTS device has a set of transmission lines (and buses) under its control. In the case of a failure, these devices recalculate the overall power redistribution and change line parameters accordingly. The exact change applied on a particular line(s) is calculated using a distributed control algorithm by communicating with other FACTS devices in the network. This way, the overall power balance of the network is properly maintained.

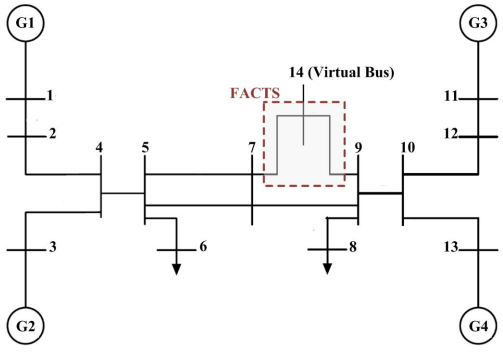


Fig. 1. A 13-Bus Test Feeder with a FACTS device

Figure 1 is a 13-bus test feeder with a FACTS device installed. The smart grid, in fact, can be regarded as the composition of numerous similar blocks.

In terms of security, it is important to maintain the confidentiality of each FACTS device's setting. If an adversary

can derive the overall state of the system, such knowledge can be used to identify the most critical and vulnerable links(transmission lines) of the network. Such cognition can be used against the power grid not only in the form of physical attacks, but also to force erroneous FACTS settings [16]; the system state may divulge sensitive operational limitations along with the present status of both FACTS devices and transmission lines.

A. Modeling the FACTS network as a DC Circuit

For the simplicity of analysis, the different interconnections of the FACTS network is compared to a DC circuit as shown in Figure 2. The ability of a FACTS device to change active power within a transmission line is similar to the capability of a variable resistor in a DC circuit. The operational limitations of a FACTS device are modeled by bounded power flow changes ($\pm 20\%$).

V. Correlation between Observability and Deducibility

The amount of sensitive information a single observer can deduce is different from what a set of collaborative observers can deduce. The strategic placement of observers is significant since readings from certain observers might turn out to be redundant. The relationship between the deducibility and the observability is an important aspect of CPS security analysis.

[10] evaluated the minimum number of observers required to fully derive all $\mathcal{D}_H$ commands in pure series and pure parallel connected networks. A similar evaluation on a network with series and parallel connections (mix connection network) resulted in Lemma 1 below.

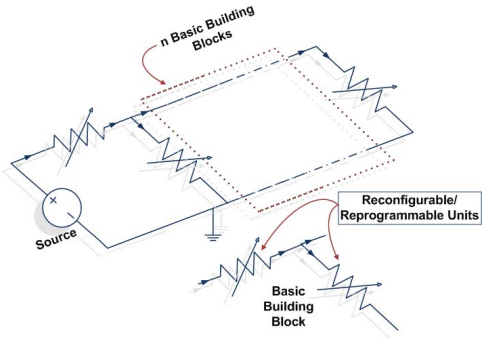


Fig. 2. The General Form of a Mix Connected Network of Reconfigurable/Reprogrammable Units

Lemma 1. ***[Minimum Number of Observers for Mix Connection Networks]*** *A mix connected network with η number of reconfigurable units and κ number of junctions can be fully deduced with a minimum of $\eta - \kappa$ number of observers.*

Lemma 1 results from progressively extending the general form of a mix connection network with the basic building block shown in Figure 2. As an example, Figure 3 shows a network of five configurable units, after inserting one basic building block.

Table I is the corresponding $\mathcal{D}_L$ observation matrix for the network in Figure 3. A $\mathcal{D}_L$ observation matrix lists changes in observations for each $\mathcal{D}_H$ action. Changes to variable resistors are denoted by $\uparrow$ and $\downarrow$ arrows.

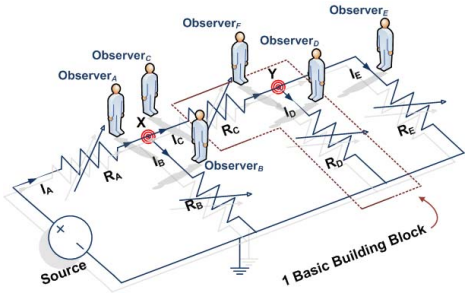


Fig. 3. A Network of Five Reconfigurable/Reprogrammable Units in a Mix Connection Network

Each row is an *execution* σ. Column 1 is the *trace* ζ and columns 2–6 is the *projection* ρ. Here is an example.

$$\sigma = \{R_E \uparrow, I_A \downarrow, I_B \uparrow, I_C \downarrow, I_D \uparrow, I_E \downarrow\}$$
$$\zeta(\sigma) = \{R_E \uparrow\}$$
$$\rho(\sigma, D_L) = \{I_A \downarrow, I_B \uparrow, I_C \downarrow, I_D \uparrow, I_E \downarrow\}$$

The objective of $\mathcal{D}_L$ observers is to uncover ζ (secret $\mathcal{D}_H$ settings) purely based on observed ρ. A single observer has less capability to do this due to his limited view of the system. For example, ρ of $observer_x$ in the second column of Table I lists that an increase in current flow could be due to any of $R_A \uparrow, R_B \uparrow, R_C \uparrow$ or $R_D \uparrow$. Thus, $observer_x$ cannot distinguish the exact command.

	$\mathcal{D}_L$ Observation				
$\mathcal{D}_H$ Change	I_A	I_B	I_C	I_D	I_E
$R_A \uparrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
$R_B \uparrow$	$\downarrow$	$\downarrow$	$\uparrow$	$\uparrow$	$\uparrow$
$R_C \uparrow$	$\downarrow$	$\uparrow$	$\downarrow$	$\downarrow$	$\downarrow$
$R_D \uparrow$	$\downarrow$	$\uparrow$	$\downarrow$	$\downarrow$	$\uparrow$
$R_E \uparrow$	$\downarrow$	$\uparrow$	$\downarrow$	$\uparrow$	$\downarrow$
$R_A \downarrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$
$R_B \downarrow$	$\uparrow$	$\uparrow$	$\downarrow$	$\downarrow$	$\downarrow$
$R_C \downarrow$	$\uparrow$	$\downarrow$	$\uparrow$	$\uparrow$	$\uparrow$
$R_D \downarrow$	$\uparrow$	$\downarrow$	$\uparrow$	$\uparrow$	$\downarrow$
$R_E \downarrow$	$\uparrow$	$\downarrow$	$\uparrow$	$\downarrow$	$\uparrow$

TABLE I
The Low level observation matrix for the network in Figure 3

In contrast, a set of collaborating $\mathcal{D}_L$ observers can build unique projections, ρ, corresponding to each ζ. As evident in Table I, there is a unique ρ for each of the 10 possible ζs. In fact, it is possible to show that a minimum of three observers can fully deduce this sample network by solving KCL equations for *junctions* X and Y.

Basic Building Blocks	Reconfigurable Units	Junctions	Minimum num. of Observers
0	3	1	2
1	5	2	3
2	7	3	4
⋮	⋮	⋮	⋮
n	$3 + 2 \times n$	$n - 1$	$n + 2$

TABLE II
THE LOW LEVEL OBSERVATION MATRIX FOR THE NETWORK IN FIGURE 3

Table II shows the relationship between the number of configurable units and the junctions. This was done by repeating the experiment with different number of basic building blocks. As a consequence, Lemma 1 can be easily proven using mathematical induction.

The result of this experiment is significant since it shows a violation of $\mathcal{D}_H$ command confidentiality due to $\mathcal{D}_L$ observations. In terms of IFPs, this is a violation of Nondeducibility security.

VI. NONDEDUCIBILITY–COMPENSATE PROPERTY FOR CPSS

Theorem 1. *[$\mathcal{P}$–**Compensate Property for CPSs**] A System of reconfigurable units with a multiplicity of non-deducible combinations has the $\mathcal{P}$–compensate property*

By definition, multiplicity in projections preserves Nondeducibility [7]. With that, consider the following two executions.

$$\sigma_1 = \{R_A \uparrow, I_A \downarrow, I_B \downarrow, I_C \downarrow, I_D \downarrow, I_E \downarrow\}$$
$$\sigma_2 = \{R_B \uparrow, I_A \downarrow, I_B \downarrow, I_C \uparrow, I_D \uparrow, I_E \uparrow\}$$

Technically, there is a potential that these two commands acting together may cancel out certain $\mathcal{D}_L$ observations, because I_C, I_D and I_E show opposite changes. The net observation resulting from conducting two simultaneous changes in R_A and R_C is shown in Figure 4.

The points at which the vertical axis crosses the horizontal axis (crossing points) in each subfigure of Figure 4 represent the combination of R'_A and R'_C values where the corresponding current reading is equivalent to the initial steady state value. This is a significant factor because at these values, there is no externally observable change. Thus, Theorem 1 is instantiated to Nondeducibility–compensate property because of the multiplicity in ρ; any two executions which include crossing points for a particular current reading have equivalent $\mathcal{D}_L$ projections.

The crossing points correspond to a particular compensating couple in the proposed framework. However, only a limited number of $\mathcal{D}_H$ events have corresponding compensating couples. This is a physical constraint of the system itself. Thus, not all $\mathcal{D}_H$ actions have corresponding corrections.

Also, from the contrapositive of Lemma (1), any number of external observers below the minimum requirement can only partially deduce the system. Event compensation achieves this by obfuscating $\mathcal{D}_L$ observations. These characteristics of the system are formalized in the following two Corollaries.

Corollary 1. *[**Partially $\mathcal{P}$–Compensatable Mix Connection Networks with Range Limited Changes**] Mix connected networks of reconfigurable units with range limited changes are partially $\mathcal{P}$–compensatable.*

Corollary 2. *[**Partially $\mathcal{P}$–Compensatable Mix Connection Networks**] Mix connected networks of reconfigurable units are partially $\mathcal{P}$–compensatable.*

A. Contribution of the Event Compensation on CPS Security

Having multiple crossing points for a particular observation gives $\mathcal{E}_S$ more options to select different action-correction pairs to prevent externally observable changes. For a state machine abstraction, this allows executions to be extended in multiple possible paths. For example, $\varphi = < R_A \uparrow, R_C \downarrow >$ or $\acute{\varphi} = < R_C \uparrow, R_A \downarrow >$, as in Figure 4(c), can be used to nullify changes in I_C. With this, the system becomes nondeterministic for $\mathcal{D}_L$ observers; absence of $\mathcal{D}_L$ observations does not reflect absence of $\mathcal{D}_H$ changes.

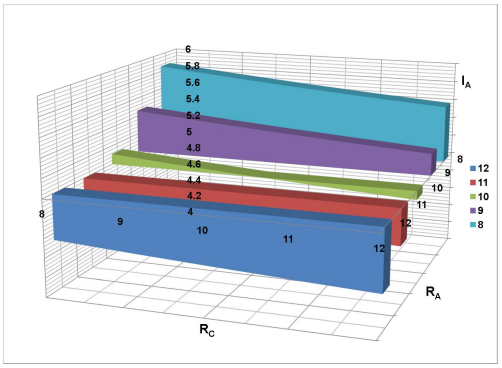
Even detected changes could be due to one of the several possible $\mathcal{D}_H$ action couples (φ or $\acute{\varphi}$ above). On top of that is the possibility of physical layer failures and interrupts. This is an important feature since it removes the uniqueness of $\mathcal{D}_L$ observations. As a consequence, such a system preserves Nondeducibility of confidential $\mathcal{D}_H$ actions[3, 10].

Equally important is the ability to recover the system from a possible failure. Having multiple possible compensating couples empowers $\mathcal{D}_H$ administrators when committing to a $\mathcal{D}_H$ command. Administrators can calculate corrections for each $\mathcal{D}_H$ command and use a compensating automata to enforce proper execution.

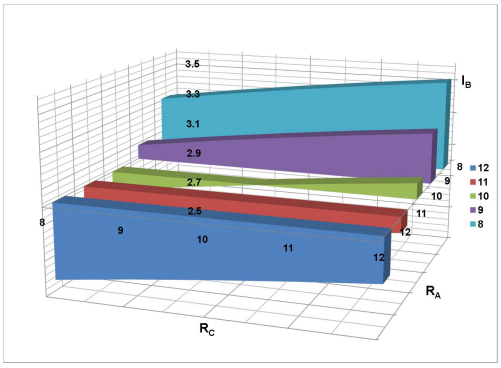
VII. CONCLUSIONS

This work formally shows how event compensation based IFPs can preserve confidentiality of actions in CPSs. The central concept in applying event compensation is to obfuscate external observations resulting from a confidential $\mathcal{D}_H$ action using two or more compensating actions. Even though the operational constraints of a system may still leave certain $\mathcal{D}_H$ actions exposed to external observation based deduction, the proposed work is an improvement over existing security models and a starting point for innovative future system security policies. The ability to compensate at least one $\mathcal{D}_L$ observation may still prove vital in preventing information flow from $\mathcal{D}_H$ to $\mathcal{D}_L$. Such compensation removes some aspects of the domain knowledge required by $\mathcal{D}_L$ observers to identify and distinguish $\mathcal{D}_H$ changes.

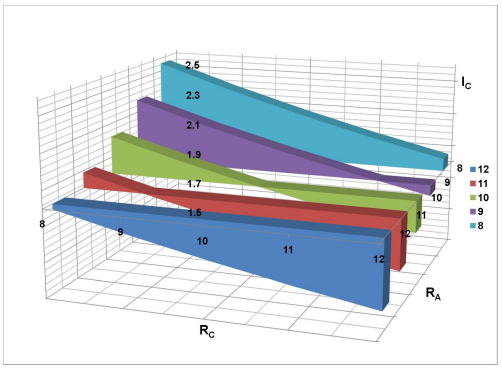
Future work expects to extend and verify the proposed framework in an actual CPS – the smart grid.



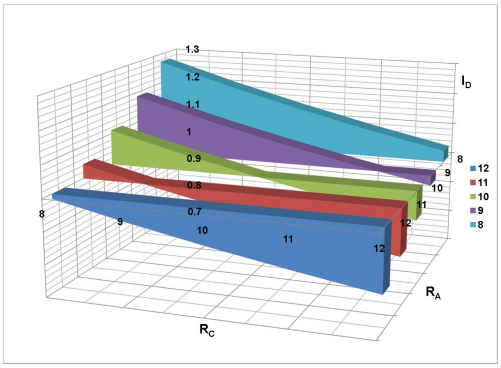
(a) The Net Effect of Changing R_A and R_C on I_A



(b) The Net Effect of Changing R_A and R_C on I_B



(c) The Net Effect of Changing R_A and R_C on I_C



(d) The Net Effect of Changing R_A and R_C on I_D

Fig. 4. The Net Effect of Changing R_A and R_C Simultaneously has on the Corresponding Changes in the Current Readings of the Network in Figure 3

REFERENCES

[1] J. Sztipanovits, "Composition of Cyber-Physical Systems," in *Procs of the 14th ECBS Workshop*, March 2007, pp. 3–6.

[2] U.S. and C. P. S. O. T. Force, "Final Report on the August 14th Blackout in the United States and Canada: Causes and Recommendations," Tech. Rep., April 2004.

[3] T. T. Gamage and B. M. McMillin, "EM Enforcing Information Flow Properties using Compensating Events," in *(HICSS-42 2009)*. IEEE CS, January 2009, pp. 1–7.

[4] A. Sabelfeld and A. Myers, "Language-based Information Flow Security," *IEEE J-SAC*, vol. 21, no. 1, pp. 5–19, Jan 2003.

[5] S. M. Bellovin and D. G. Conway, "Security as a Systems Property," *IEEE SP*, vol. 7, no. 5, p. 88, 2009.

[6] F. B. Schneider, "Enforceable Security Policies," *ACM TISSEC*, vol. 3, no. 1, pp. 30–50, 2000.

[7] J. McLean, "A General Theory of Composition for a Class of "Possibilistic" Properties," *IEEE TSE*, vol. 22, no. 1, pp. 53–67, 1996.

[8] N. Nagatou and T. Watanabe, "Run-Time Detection of Covert Channels," in *ARES '06*, 2006, pp. 577–584.

[9] B. Alpern and F. B. Schneider, "Defining Liveness," Cornell University, Tech. Rep., 1984.

[10] T. T. Gamage and B. M. McMillin, "Observing for Changes: Nondeducibility Based Analysis of Cyber-Physical Systems," in *Proceedings of the 3rd (IFIP WG 11.10)*, Hanover, NH, April 2009, pp. 169–183.

[11] H. Mantel, "Possibilistic Definitions of Security – An Assembly Kit," in *IEEE CSFW-13*, 2000, pp. 185–199.

[12] T. Amtoft and A. Banerjee, "A Logic For Information Flow Analysis With An Application To Forward Slicing Of Simple Imperative Programs," *Science of Computer Programming*, vol. 64, no. 1, pp. 3–28, 2007.

[13] K. W. Hamlen, G. Morrisett, and F. B. Schneider, "Computability Classes for Enforcement Mechanisms," *ACM TPLS*, vol. 28, no. 1, pp. 175–205, 2006.

[14] A. Zakinthinos and E. S. Lee, "A General Theory of Security Properties," in *IEEE SP*. Washington, DC, USA: IEEE CS, 1997.

[15] J. Ligatti, L. Bauer, and D. Walker, "Run-time Enforcement of Nonsafety Policies," *ACM TISSEC*, vol. 12, no. 3, Jan 2009.

[16] A. Faza, S. Sedigh, and B. McMillin, "Reliability Analysis for the Advanced Electric Power Grid: From Cyber Control and Communication to Physical Manifestations of Failure," in *(SAFECOMP'09)*, Sep 2009, pp. 257–269.