Chapter 12

USING INFORMATION FLOW METHODS TO SECURE CYBER-PHYSICAL SYSTEMS

Gerry Howser

Abstract The problems involved in securing cyber-physical systems are well known to the critical infrastructure protection community. However, the diversity of cyber-physical systems means that the methods used to analyze system security must often be reinvented. The issues of securing the physical assets of a system, the electronics that control the system and the interfaces between the cyber and physical components of the system require a number of security tools. Of particular interest is preventing an attacker from exploiting nondeducibility-secure information flows to hide an attack or the source of an attack. This potentially enables the attacker to interrupt system availability.

This chapter presents an algorithm that formalizes the steps taken to design and test the security of a cyber-physical system. The algorithm leverages information flow security techniques to secure physical assets, cyber assets and the boundaries between security domains.

Keywords: Cyber-physical systems, information flow security, nondeducibility

1. Introduction

Modern critical infrastructures, which comprise computers, embedded devices, networks and software systems, are vital to day-to-day operations in every sector [6, 28]. A prominent feature of a cyber-physical system (CPS) is that it consists of embedded computers and communications networks that govern physical manifestations and computations, which, in turn, affect how these two major components interact with each other and the outside world [17]. The combined discrete computational and continuous physical nature of a cyber-physical system compounds the difficulty; not only do these systems have different semantics, but the system boundaries are blurred from pure cyber and pure physical systems. The lack of clearly-defined boundaries creates new se-

M. Rice, S. Shenoi (Eds.): Critical Infrastructure Protection IX, IFIP AICT 466, pp. 185–205, 2015.
DOI: 10.1007/978-3-319-26567-4_12

curity and privacy vulnerabilities, some of which are difficult to detect and correct.

A textbook definition of computer security [4] relies on three fundamental concepts: confidentiality, integrity and availability. In the case of critical infrastructure protection, a unique problem exists from a security perspective – changes in the physical portion of an infrastructure are observable, which inherently violates confidentiality in the infrastructure. Adversaries can potentially derive sensitive internal settings by observing external system changes. This derived knowledge, coupled with the semantic knowledge of the system, can be used against the system in the form of integrity and availability attacks.

The problem of securing physical assets is as old as society itself. The idea of more secure zones completely contained within less secure zones is still one of the most effective ways to secure a physical area. This concept can be adapted to designing security for electronics, computer systems and the cyber portions of cyber-physical systems. Access control methods combined with methods that divide the cyber assets into domains with varying levels of security [2] can produce a reasonably secure cyber system.

However, cyber-physical systems present unique challenges in terms of security. Not only must the cyber and physical components of the system be secured, but the two sets of components are interdependent. This interdependence or coupling leads to complex situations that cannot usually be described by traditional security models. Cyber-physical system security must also address the flow of information between the cyber and physical systems as well as information leaked by the act of observing the physical system. The security domains are intricate in that more secure layers are not completely contained within less secure layers (like an onion) and the security domains frequently overlap in highly complex ways.

What is lacking is a clear methodology for securing cyber-physical system assets despite their inherent complexity. This chapter presents an algorithm that can guide a team of experts who are familiar with a specific cyber-physical system to develop descriptions of physical asset security, cyber asset security, and secure information flows between the various assets. The question is: How can all the parts of a cyber-physical system that cannot be hidden from an observer be secured? Due to its very nature, a cyber-physical system leaks some information that an adversary can use against the system. Unfortunately, it is not enough to secure the physical and cyber systems independently; it is imperative to also secure the flow of information between the two systems.

2. Background

This section discusses the principal concepts involved in using information flow methods to secure cyber-physical systems. Table 1 describes the nomenclature used in this chapter.

Table 1. Nomenclature.

Symbol	Description
s_x	Boolean state variable; s_x is true or false
$\neg$	Logical NOT
$\wedge$	Logical AND
$\vee$	Logical OR
$\oplus$	Exclusive OR: $\varphi \oplus \psi \equiv (\varphi \vee \psi) \wedge \neg(\varphi \wedge \psi)$
$\rightarrow$	Material implication: $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
φ	Arbitrary logical formula or statement evaluated in $w \in W$
ψ	Arbitrary logical formula or statement evaluated in $w \in W$
wff	Well-formed logical formula
$\Box\varphi$	Modal "must be so" operator ("it is always that…")
$\Diamond\varphi$	Modal "possible" operator ("it is possible that…")
$w \vdash \varphi$	Statement φ is valid in world w ("yields")
$w \models \varphi$	Values from world w cause φ to evaluate to true ("models")
$\mathfrak{F}$	Kripke frame
$\mathfrak{M}$	Kripke model built over a Kripke frame
${}_wR_{w'}$	Transition function from world w to $w' : w, w' \in W$
$\mathfrak{R}$	Set of transition functions in a complete Kripke frame
$\mathbb{V}_x^i(\varphi)$	Valuation function of Boolean x in domain i
$B_i\varphi$	Modal BELIEF operator
$I_{i,j}\varphi$	Modal INFORMATION TRANSFER operator
$U_j\varphi$	Modal UTTERANCE (broadcast) operator
$T_{i,j}$	Modal TRUST operator

2.1 Information Flow Security

Two main approaches are available within the context of critical infrastructure system security policies and mechanisms: (i) access control methods; and (ii) information flow methods. Although access control methods are frequently used and are well understood, considerable evidence in the literature suggests that information flow models are the most promising mechanisms for cyber-physical systems [5, 11, 20–23, 26, 29].

The principal focus of information flow security is to prevent unintended high-level (secure/private) domain information disclosures to a low-level (open or public) domain. A security policy may define exactly what low-level users are forbidden to know about a high-level domain, but the enforcement depends on sound implementation and trust. Traditional security models such as the Harrison-Ruzzo-Ullman (HRU) model [12], Bell-LaPadula (BLP) model [1, 2] and Biba model [3] restrict access to information and resources by taking an access control approach to the security problems of assuring confidentiality, integrity and availability.

Access control methods are unable to address the issues of indirect security violations that involve explicit or implicit information flows [13]. Imposing

strict rules and restrictions on the assets that can be accessed cannot govern how the assets and data are used after they have been accessed. A high-level process can easily signal information to a low-level process by setting a flag, filling a buffer or writing data to a non-secure location. While this is an indirect violation of the Bell-LaPadula *-property [2], no direct violation of the "write-down" property occurs; yet, the low-level process may have gained sensitive information that resides in a high-level security partition. Access control methods cannot enforce proper security on information flows (nor were they designed to do so) without enforcing severe restrictions on the access to sensitive assets. This issue has been pointed out by many researchers [5, 11, 20–23, 26, 29]. Using access control models to correctly describe and constrain information flows is attempting to do something that they were never designed to do. These models trust that all agents act according to the rules; they cannot guarantee security if an agent is not trustworthy.

While access control models rely heavily on idealized security partitions, cyber-physical systems usually have less well-defined and well-ordered partitioning. A more useful approach is to use information flow security policies to impose restrictions on all known information flows between security partitions with less regard for the high-level/low-level ordering. In short, information flows between security partitions should be viewed from the perspective of various loosely-defined security domains with less regard for security clearances. In information flow security, a violation by an agent in a lower security domain could involve actions such as monitoring an information flow, disrupting an information flow, modifying an information flow or simply detecting the existence of an information flow. The only solution is to prevent information flows between processes that are not allowed to communicate under the security policies of the system [8, 29]. Many information flow models describe these desired and unwanted information flows. Some of the most notable models are noninterference [10], noninference [24], nondeducibility [27] and multiple security domain nondeducibility (MSDND) [14].

The physical assets of a critical cyber-physical system must be thought of as low-level outputs because the assets can be physically viewed by an attacker. Frequently, by monitoring the changes in physical assets, sensitive information about the states and actions of a system can flow to the attacker. These flows can be described using the notions of noninterference, noninference and nondeducibility. Noninterference is most consistent with the popular notion of high security. If a system is noninterference secure, agents in the low level are unaware of any actions in the high level. Noninference-secure systems prevent low-level agents from determining if the actions they observe are due to high-level or low-level events. Nondeducibility-secure systems prevent all low-level agents from determining high-level events no matter how many low-level events are observed. In essence, in a nondeducibility-secure system, low-level events might be the result of many different high-level, or even low-level, events and, therefore, no reliable information is leaked.

2.2 Physical System Security

Fortunately, it is known how to secure physical assets. This has likely been done since before the dawn of recorded history. In essence, securing a physical system such as an electric grid is the same problem as securing any other physical asset. No matter how secure the system is electronically, an attacker who gets to the system can disable it.

2.3 Cyber System Security

While tight physical security is ideal, it makes a cyber-physical system more difficult for users to access. Ideally, a mobile user would access the system only via a network. Indeed, with the advent of cloud computing, the user need not even know where the physically-secured assets are located. Do you really know where your email server is located? Do you know where your Internet service provider connects to the Internet?

Over the past thirty or forty years, an excellent suite of cyber security tools has been developed. It is known how to secure a cyber system and the trade-offs between security and accessibility are well understood. Techniques such as user names and passwords, when combined with common sense and encryption, work well to manage the use of assets. Messages between cyber systems can be kept private by the proper use of encryption in most cases. Indeed, cyber security is almost as well understood as physical security.

2.4 Information Flow as Information Leakage

Too many people believe that combining physical security with cyber security is adequate to secure a cyber-physical system. However, it is not that simple. A cyber-physical system typically leaks information when it is operating normally. For example, many modern cars automatically unlock when the correct key fob is present. The information that the correct key fob is within range is leaked to the driver and to any observers in the vicinity. If an individual enters a locked car without physically unlocking the door, it is safe to assume that the individual has the correct key fob somewhere on his or her person. If not, the door would remain locked. Secure information that "the individual has the key" is leaked by way of noninterference [10, 13, 21]. An information flow has occurred from the secure domain of the car to the unsecured domain of the outside world. Such flows of information could have serious consequences in mission-critical systems.

What is needed is a method – preferably an algorithm – to ensure that a cyber-physical system is properly secured from physical threats and cyber threats, and that all information flows are either eliminated or secured. Algorithm 1 formalizes the steps needed to design and test the security of a cyber-physical system.

Algorithm 1 : Cyber-physical system security.

```
1: procedure CPS Security(status)
2:     Set status to insecure                                  ▷ Assume the worst
3:     while status insecure do                                ▷ Check physical security
4:         procedure Physical Security(status)
5:             Disallow all access                             ▷ Start with no access
6:             Allow limited access          ▷ Only allow as much access as needed
7:             Check known physical threats
8:             if physically protected then
9:                 Set status to secure
10:            else
11:                Refine physical security
12:            end if
13:        end procedure
14:        if status secure then
15:            procedure Cyber Security(status)
16:                Set status to insecure
17:                Describe CPS using HRU                      ▷ Access control model
18:                if HRU secure then
19:                    Describe CPS using BLP/Lipner           ▷ Similar to "Top Secret"
20:                    if BLP/Lipner secure then
21:                        Set status to secure
22:                    else
23:                        Refine BLP/Lipner security
24:                    end if
25:                else
26:                    Refine access control matrix security
27:                end if
28:            end procedure
29:        end if
30:        if status secure then
31:            procedure MSDND Security(status)
32:                Set status to insecure
33:                while status insecure do
34:                    Describe CPS using MSDND
35:                    Test all known information flow threats
36:                    if MSDND secure then
37:                        Set status to secure
38:                    else
39:                        Refine MSDND security
40:                    end if
41:                end while
42:            end procedure
43:        end if
44:    end while
45: end procedure
```

3. Securing Cyber-Physical Systems

Because a cyber-physical system naturally divides into three main areas (physical system, cyber system and the interactions between the two systems), the cyber-physical system can be secured in three major steps with some adjustments. The steps are really loops in the sense that it is possible to go back to the start frequently because the fundamental understanding of the cyber-physical system may change during the security analysis. For example, it may be determined that some objects are really subjects or that some security domains may have to be reconsidered. Changes made to secure one cyber asset may expose previously-secure cyber assets to attack.

3.1 Physical Security Analysis

The first step in securing a cyber-physical system is to implement physical security. Without physical security there is no point in attempting any cyber security. Physical security can be achieved by following the procedure CPS Security in Algorithm 1. As in the case of configuring a firewall (network firewall or physical firewall), it is best to start by eliminating all access. Should an access path be overlooked, it will have already been closed. After the assets are physically secured, each user is analyzed to see exactly what physical access the user requires in order to use the cyber-physical system. Usually, only the engineering and maintenance staff would require physical access to the cyber-physical system, but there will always be some individuals who think they require the ability to physically touch the equipment. A good rule of thumb is to view any individual with physical access as a physical threat to the system; this includes security personnel as well.

3.2 Cyber Security Analysis

Several tools have been developed for securing the cyber assets of cyber-physical systems. This work will touch on a few that are useful for conducting information flow security analyses. It is still critical to secure the electronics, computing equipment, data and communications channels, but the main thrust is to concentrate on the methods that can help identify the parts of a cyber-physical system that are components of an information flow security analysis.

- **Harrison-Ruzzo-Ullman Access Control Matrix:** The access control matrix introduced by Harrison et al. [12, 21] is analogous to the common permissions in Unix and Linux systems. The Harrison-Ruzzo-Ullman model separates the entities under consideration into two categories that are of interest in an information flow security study. One category comprises objects that can only react to commands and report status, but not initiate any actions. The other category comprises subjects that can initiate actions that affect other subjects or objects, in addition to doing all the things that an object can do.

Table 2. Harrison-Ruzzo-Ullman model and Linux analog.

	HRU Object	**HRU Subject**
Linux `r`	Allows object to report information	Same as for HRU object
Linux `w`	Allows object to accept commands	Same as for HRU object
Linux `x`	Not allowed	Subject issues commands

A Linux analogy is useful (Table 2). In Linux, everything is treated as a file and the permissions on the file determine what the file is allowed to do. A file with the `x` (execute) permission set is analogous to a subject and can initiate actions or be acted upon by other subjects. If the `x` permission is not set, the file cannot initiate any actions, but can only be acted upon, much like a Harrison-Ruzzo-Ullman object. Unfortunately, the Harrison-Ruzzo-Ullman model is not of much use in describing a cyber-physical system or even in examining the cyber components for vulnerabilities. However, it can be used to quickly identify the entities of the system and assign each entity the role of a subject or object, which are important when performing an information flow security analysis.

- **Bell-LaPadula Model:** The Bell-LaPadula model [1, 21] describes the protection of the confidentiality, integrity and availability of assets by assigning each subject or object to an appropriate security domain. The Bell-LaPadula model also forces an analyst to order the security domains from least secure to most secure; while this is highly appropriate for cyber assets, it is not always appropriate for cyber-physical systems. The domains that result when using the Bell-LaPadula model to describe the system are usually a good starting point for determining the security domains for a multiple security domain nondeducibility (MSDND) information flow analysis.

- **Biba and Lipner Models:** If the security partition analysis performed on a system specified using the Bell-LaPadula model leads to issues related to the trust and integrity between partitions, further granularity may be introduced by using the Biba [3] or Lipner [19] models. The two models allow a form of lattice security, but, unfortunately, while a cyber-physical system may be described in a better manner, it is usually not more secure. The Biba and Lipner models were developed to secure purely cyber assets and attackers have many simple methods to get around the constraints of the models.

3.3 Complications

Complications occur when a cyber system controls a physical system. As noted above, the earlier security models were developed to handle pure cyber

assets. However, cyber-physical systems introduce novel vulnerabilities for the following reasons:

- Physical assets can be observed by an attacker.
- Physical and cyber assets are inextricably intertwined and can be exploited easily.
- Cyber-physical systems always leak some information. If detailed information flow security analyses are not performed, then the cyber-physical systems may be highly vulnerable.

For example, if a large data center is hidden underground, an attacker might still find the center by observing the cooling system which is invariably outdoors. Likewise, the security gates and traffic into and out of the center are difficult to hide.

As another example, assume that a cluster of Linux processors is set up to crack the encryption of Internet traffic. While it is relatively simple to remove the rights of a casual employee to monitor the processes on the cluster, it is difficult to mask the changes in the amount of processing that occur at any given time. The high usage of cyber assets leads to an observable increase in the use of physical assets such as fans, power, cooling and blinking lights.

Drive-by-wire automobiles, fly-by-wire airplanes, subway traffic control systems and smart traffic lights can be observed over long periods of time. Their actions cannot be hidden and lead to information about system commands and responses being leaked. If the flows are identified, they may be masked as proposed by Gamage [9] or at least be monitored for possible attacks.

4. Nondeducibility and Security

Several information flow security models, such as noninterference and noninference [8, 10, 21], were developed in the 1980s and 1990s to protect information flow between cyber processes. Flows that can be described accurately by these early models can also be described by multiple security domain nondeducibility (MSDND); this reduces the number of models that need to be considered. Nondeducibility of an information flow is important because, not only can information be leaked without detection, but an attacker can use nondeducibility to hide an attack completely or at least hide the source of the attack.

4.1 Deducibility vs. Nondeducibility

Assume that a network is using a hardware encryption block for all communications. If the hardware encryption block does not draw power when it passes a plaintext message, an adversary would be able to correctly deduce the fact that a message is encrypted when extra power usage is observed. In this case, the encryption of a message is deducible.

On the other hand, if the hardware encryption block were to be changed to encrypt all messages and then send either encrypted or plaintext messages,

the power usage would be essentially the same for an encrypted message and a plaintext message. An adversary could not be able to correctly deduce if a message was being sent as encrypted or as plaintext. In this case, the state of the message is nondeducible or nondeducibly secure.

4.2 Definitions

This section provides the definitions of several key concepts.

Kripke Frames and Models. This work examines information flow security from the viewpoint of Kripke frames and models [14]. It is enough to know that a frame is made up of worlds ($w \in W$ where each world is a unique combination of binary state variables) and the transitions between worlds. A model with valuation functions to evaluate logical questions about the state variable values for a world can be built on the Kripke frame. This model can then be used to describe the information flows and state changes of a cyber-physical system. It is important to note that, if there is no valuation function for a state variable for a given world, then neither the value of the state variable nor the truth value of any query containing the variable can be determined.

This work uses the shorthand $\mathbb{V}^i_\varphi(w)$ to denote the valuation function used by an agent i for a logical query with a number of state variables. If the entity has valuation functions for all the state variables in query φ, then the functions can be composed into a single valuation for the query on the world w.

Nondeducibility. Multiple security domain nondeducibility (MSDND) [14] was proposed to address situations where information flows in cyber-physical systems are difficult to describe using existing models. If two states are mutually exclusive, i.e., one and only one state may be true and the other must be false, and an agent cannot evaluate either state, then the resulting states are nondeducibility secure. Simply put, if the state of a true/false variable cannot be deduced, then the variable is nondeducibility secure.

Multiple Security Domain Nondeducibility (MSDND). MSDND is defined over a set of state changes of a cyber-physical system. Suppose $x_i \in X$ is a set of state variables. In order to know the state of any $x_i \in X$, the model must have some valuation function that returns the truth value of the variable. This is denoted by $\mathbb{V}^j_i(w)$, which returns the truth value of x_i as seen by entity j. It is entirely possible that one entity can evaluate x_i and another cannot. For example, a systems administrator may be able to see that file `fred` exists while user Sam might not. Suppose $\varphi = x_i$ is a logical expression. Then, the expression must be either true or false on every world; there is no other choice.

Therefore, φ is MSDND secure on world w if and only if Equation (1) or Equation (2) is true as in Figure 1. In other words, if for every possible combination of events, an entity cannot evaluate a logical expression, then the expression is MSDND secure.

$$MSDND = \forall w \in W : w \vdash \Box\,[s_x \oplus s_y] \wedge \Big[w \models (\not\exists \mathbb{V}^i_x(w) \wedge \not\exists \mathbb{V}^i_y(w))\Big] \quad (1)$$

When s_x is $\varphi = \top$ and s_y is $\neg\varphi = \top$:

$$MSDND = \forall w \in W : \Big[w \models (\not\exists \mathbb{V}^i_{\varphi}(w))\Big] \quad (2)$$

Figure 1. Formal definition of MSDND.

5. Nondeducibility Secure Attacks

This section discusses two examples of nondeducibility secure attacks, one involving a drive-by-wire car and the other involving a cream separator at an ice cream plant. As a matter of fact, the modeling and analysis of the cream separator formally shows how the Stuxnet virus operated [15].

Table 3. HRU analysis of the drive-by-wire car.

Entity	HRU	Actions
car	Subject	Can initiate and respond to actions
corp	Subject	Can initiate and respond to actions
driver	Subject	Can initiate and respond to actions
tc	Object	Can respond to commands

5.1 Drive-by-Wire Car

A drive-by-wire car equipped with remote assistance such as OnStar or Toyota Connect is a good example of a cyber-physical system with complex security domains [13]. The model comprises a (*corp*) that provides an automobile (*car*) with onboard drive-by-wire functionality and traction control (*tc*) and service to the driver of the automobile (*driver*) in the form of remote assistance (navigation, remote unlock, remote shutdown, etc.). Table 3 shows the results of a Harrison-Ruzzo-Ullman analysis of the system.

Table 4 presents the results of a Bell-LaPadula (BLP) analysis, which yields increasingly secure domains with *tc* being the most secure. However, these results do not reflect the reality of the drive-by-wire car. As will be shown later, in some modes of operation, the driver is unable to issue commands and essentially becomes a passenger in the car [13]. In reality, the security domains are overlapping and complex (Figure 2). What is more, the relationships between the *car*, *corp*, *driver* and *tc* change depending on the mode of operation. The Bell-LaPadula model is simply not designed to handle this situation.

The security of this cyber-physical system depends on more than access control or security domains. The operation of the car leads to complex information

Table 4. Bell-LaPadula domains of the drive-by-wire car.

Entity	BLP Domain	Security Level
driver	SD^{driver}	4
car	SD^{car}	2
corp	SD^{corp}	3
tc	SD^{tc}	1

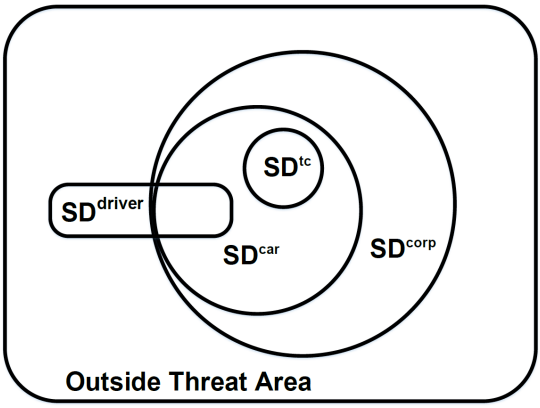

Figure 2. Security domains of the car.

flows between the entities and these flows must be secured from interruption and modification. The MSDND model was developed to accurately describe such information flows. Some flows may need to be made MSDND secure from outside observation; these flows can be made even more secure. Other flows may be critical to the operation of the car and open to an MSDND-secure attack that allows an attacker to hide his actions or at least hide the source of the attack. In this case, steps should be taken to make the information flow not MSDND secure using a separate physical indicator or measurement.

The following three modes of operation are relevant to the discussion:

- **Normal Operation:** The *driver* can operate the *car*. From this, the *driver* knows he/she controls the *car*.
- **Hazardous Road Conditions:** Most modern automobiles are equipped with traction control systems that automatically correct when there is a loss of traction. When traction control (*tc*) is active, the *car* will attempt to correct a skid and counter anything the *driver* does that would make the skid worse.
- **Corporate Remote Operations:** If the *car* is equipped with a service such as OnStar, the corporation (*corp*) can issue commands to the *car*.

> The *driver* must trust the corporation to act in his best interests [13, 14]. Television commercials present the benefits of access to a car via a network such as the Internet or a corporate connection, but is this a good thing? It may be fun to lock or unlock car doors from a cell phone [30], but what happens when the cell phone is hacked or stolen? What if the corporation or car network is hacked and the hacker decides to simply power off as many cars as possible [25]?

The fundamental question is: Who is in control when the *car* refuses to respond? It is obvious that the *car* will only respond to one set of commands from either the *driver*, traction control (*tc*) or the corporate network (*corp*). Depending on which mode the *car* is in, the *driver* may be unable to distinguish who or what is actually in control. Of particular interest is remote operation by *corp*, which exists in one security domain, versus operation by *driver*, which is in another security domain (see Figure 2). What the *driver* can and cannot ascertain is governed by the information flow that exists between domains – the cyber domain as well as the physical domain. The ensuing discussion shows how classical models of information flow and deducibility break down in the cyber-physical environment.

If traction control takes control of the *car*, the *driver* notices a complete lack of response to *driver* commands. While this is disconcerting at first, the traction control (*tc*) reacts to more accurate and timely knowledge of the road conditions and any lack of traction by the tires than the *driver*.

But what if someone uses the network to take control of the car? In this case, the *car* will not respond to *driver* commands, but will only respond to the network commands.

There are two questions of interest here. First, who is in control of the *car*? Second, can the *driver* correctly deduce who is in control?

Obviously, the *car* will only respond to commands from one source at a time. The highest level of commands are from the corporation network (*corp*) such as OnStar, then the traction control module (*tc*) and finally the *driver*. If the commands come from the *driver*, all is well and the *driver* can correctly deduce who is in control. However, the *driver* sees the same loss of control and unexpected actions if either *tc* or *corp* is in control, but cannot determine which entity is in control.

Mathematically, the question of who is in control of the car is expressed as:

$$\forall w \in W : w \vdash [tc \oplus corp] \wedge \left[\not\exists \mathbb{V}_{corp}^{driver}(w) \wedge \not\exists \mathbb{V}_{tc}^{driver}(w) \right] \tag{3}$$

This is exactly what is required to show that the control of the *car* is MSDND secure from the *driver*. In this case, MSDND has been turned against the *driver* to hide a possible attack. Moreover, because a failure of the *tc* would act in the same way, the source of the attack is also MSDND secure by the same reasoning. Indeed, it is quite possible that, in the case of a cyber-physical system, security tools and methods may be turned against the system itself. In

Table 5. Harrison-Ruzzo-Ullman analysis of the cream separator.

Domain	Name	Type
0	Cream Separator	Object
1	Virus	Subject
2	Controller	Subject
3	Monitor System	Subject
4	Human Operator	Subject

fact, when attacking a cyber-physical system, it is often enough to disrupt the normal flow of commands to damage or destroy the system.

5.2 Cream Separator

Assume that an ice cream company has discovered the exact butterfat content to make perfect ice cream. The process employs a centrifuge connected to a programmable logic controller (PLC) to act as a cream separator.

Table 6. Process control system security domains.

Domain	Valuation	Name
SD^0	$\mathbb{V}^0$	Cream Separator
SD^1	$\mathbb{V}^1$	Virus
SD^2	$\mathbb{V}^2$	Controller
SD^3	$\mathbb{V}^3$	Monitor System
SD^4	$\mathbb{V}^4$	Human Operator

A cursory Harrison-Ruzzo-Ullman analysis results in the identification of subjects and objects shown in Table 5. However, an attempt to describe the system using the Bell-LaPadula model produces a less than satisfactory set of security domains. The cream separator does not easily divide into the security partitions expected when using the Bell-LaPadula model. The security domains are not hierarchical as would be expected, although there is a logical structure, the Bell-LaPadula model simply fails for this system. However, the Bell-LaPadula model does assist in decomposing the cyber-physical system into security domains. As such, there are five separate security domains as defined in Table 6 and illustrated in Figure 3. In this case, an information flow security analysis yields a more useful description.

The plant can use an MSDND-security analysis to spot this possibility where other methods will not. At the separator, the speed is not MSDND secure because the sensors on the separator correctly read the speed. However, when the speed reading is reported to the controller, the virus intercepts the reading

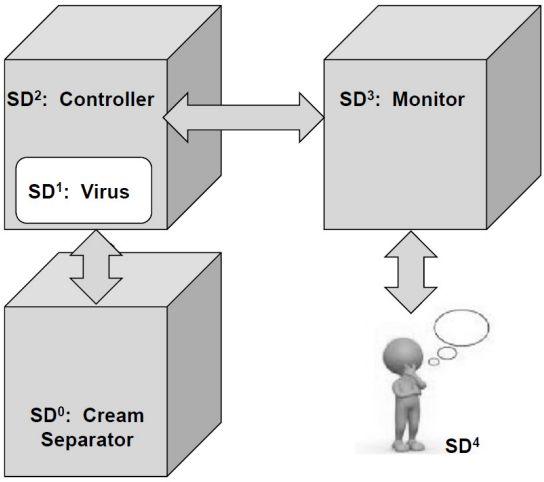

Figure 3. Abstract plant model with security domains.

and reports to the controller that the separator is operating at the correct speed regardless of its actual speed. Likewise, the virus intercepts messages from the controller to speed up or slow down the separator. Eventually, the separator spins at the wrong speed and poor quality ice cream is produced. To assist in this analysis, Liau's BIT logic [18] presented in Table 7 is employed. Interested readers are referred to [15] for the complete axiomatic system.

The separator makes cream under the control of a programmable controller. A monitor reads what the controller senses from the physical process and displays the results to a human. Consider a portion of the controller to be a program that is not functioning as anticipated. This could be due to many reasons ranging from a software bug to an actual virus. In this model, multiple security domains exist, but the notions of high and low security are not relevant. In the remainder of the discussion, let φ denote "cream is being separated properly." Obviously, either φ or $\neg\varphi$ must be true at all times. Under normal conditions, the controller/monitor system oversees the cream separator and makes adjustments to ensure that φ is true.

If the system is operating correctly (without the virus), all operations performed by the controller are successfully carried out and reported back to the monitor deducibly. In other words, every action in a domain is uniquely identifiable in another domain; the cream separation process is reported correctly to the operator, operator commands are carried out by the controller on the cream separator, etc.

The key to demonstrating the integrity of the system is to show that the desired information flow cannot be disrupted. If the system is not MSDND secure, then each observation or command can be uniquely attributed to its corresponding command and observation. To start with, it is necessary to

Table 7. BIT logic axiomatic system.

1. Definitions of Logical and Modal Operators	
D6:	$B_i\varphi$ Entity i believes the truth of φ
D7:	$I_{i,j}\varphi$ Entity j informs i that $\varphi \equiv \top$
D8:	$T_{i,j}\varphi$ Entity i trusts the report from j about φ
2. Axioms	
P:	All the tautologies from the propositional calculus
B1:	$[B_i\varphi \wedge B_i(\varphi \rightarrow \psi)] \rightarrow B_i\psi$
B2:	$\neg B_i\bot$
B3:	$B_i\varphi \rightarrow B_iB_i\varphi$
B4:	$\neg B_i\varphi \rightarrow B_i\neg B_i\varphi$
I1:	$[I_{i,j}\varphi \wedge I_{i,j}(\varphi \rightarrow \psi)] \rightarrow I_{i,j}\psi$
I2:	$\neg I_{i,j}\bot$
C1:	$B_iI_{i,j}\varphi \wedge T_{i,j}\varphi \rightarrow B_i\varphi$
C2:	$T_{i,j}\varphi \equiv B_iT_{i,j}\varphi$
3. Rules of Inference	
R5:	From $\vdash \varphi \equiv \psi$ infer $\vdash T_{i,j}\varphi \equiv T_{i,j}\psi$
4. Logical Statement Formulation Rules	
F1:	If φ is a wff, so are $\neg\varphi, \Box\varphi,$ *and* $\Diamond\varphi$
F2:	If φ and ψ are wff, so is $\varphi \vee \psi$
F3:	If φ and ψ are wff, so is $\varphi \wedge \psi$
F4:	If φ is a wff, so are $B_i\varphi$, and $\neg B_i\varphi$
F5:	If φ is a wff, so are $I_{i,j}\varphi$, and $\neg I_{i,j}\varphi$
F6:	If φ is a wff, so are $T_{i,j}\varphi$, and $\neg T_{i,j}\varphi$

establish that the cream separator correctly reports its status and dutifully follows the commands sent to it by the controller.

Using BIT logic [18], information transfer is represented by $I_{i,j}\varphi$, which clearly states how i gains knowledge of φ (see Table 7). The information transfer operator inherently assumes j will not lie to i; however, this restriction allows liars to lie to trusting agents. Note that the information is transferred directly to an agent who has no direct way to evaluate whether or not j is a liar. As such, the status of the cream is not MSDND secure at the cream separator. The physical cream separator correctly reports the status of its cream.

Theorem 1. *The cream status is not MSDND secure at the cream separator.*

Proof. Clearly, $(\varphi \oplus \neg\varphi) = \text{true}$, so the first condition for MSDND is met by the definition. However, the separator directly measures the cream and, therefore,

both $\mathbb{V}^0_{\varphi}(w)$ and $\mathbb{V}^0_{\neg\varphi}(w)$ are correctly evaluated for any w and the conditions for MSDND are not met. □

In order to cause the maximum amount of disruption, the virus can completely block information flow from the cream separator to the controller. However, this is a trivial case that is easily detected via timeouts on readings from the cream separator. Instead, the focus is on the more insidious case where the virus fabricates readings to create a false information flow. The fabricated readings cause an observation at the monitor to be consistent with multiple possibilities in the physical system, essentially making the system nondeducible from the perspective of the human operator.

Theorem 2. *The speed of the separator is MSDND secure for SD^2 during the attack phase for the infected systems and any agent i in SD^2 will believe all is well or φ.*

Proof. By definition, $(\varphi \oplus \neg\varphi) = \text{true}$, so the first condition for MSDND is met. If φ cannot be correctly evaluated in SD^2, then both conditions are met.

Case (i): Separator speed is nominal and $\varphi = \text{true}$

1.	φ	Separator speed is nominal
2.	$w \vDash \mathbb{V}^0_{\varphi}(w) = \text{true}$	Definition of $w \vDash \mathbb{V}^0_{\varphi}(w)$
3.	$I_{1,0}\varphi$	Sensor reports to virus
4.	$B_1 I_{1,0}\varphi$	Virus believes sensor report
5.	$T_{1,0}\varphi$	Virus trusts the sensors
6.	$B_1 I_{1,0}\varphi \wedge T_{1,0}\varphi \rightarrow B_1\varphi$	Axiom C1, Virus believes status
7.	$I_{2,1}\varphi$	Virus reports all is well
8.	$B_2 I_{2,1}\varphi$	PLC believes interface report
9.	$T_{2,1}\varphi$	PLC trusts reports
10.	$B_2 I_{2,1}\varphi \wedge T_{2,1}\varphi \rightarrow B_2\varphi$	Axiom C1, PLC believes φ
11.	$w \vDash \mathbb{V}^2_{\varphi}(w) = \text{true}$	$\mathbb{V}^2_{\varphi}(w)$ always returns true

Case (ii): Separator speed is not nominal and $\neg\varphi = \text{true}$

1.	$\neg\varphi$	Separator speed is not nominal
2.	$w \vDash \mathbb{V}^0_{\varphi}(w) = \text{false}$	Definition of $w \vDash \mathbb{V}^0_{\varphi}(w)$
3.	$I_{1,0}\neg\varphi$	Sensor reports problem to virus
4.	$B_1 I_{1,0}\neg\varphi$	Virus believes sensor report
5.	$T_{1,0}\neg\varphi$	Virus trusts the sensors
6.	$B_1 I_{1,0}\neg\varphi \wedge T_{1,0}\neg\varphi \rightarrow B_1\neg\varphi$	Axiom C1, Virus believes status
7.	$I_{2,1}\varphi$	Virus reports all is well
8.	$B_2 I_{2,1}\varphi$	PLC believes interface report
9.	$T_{2,1}\varphi$	PLC trusts reports
10.	$B_2 I_{2,1}\varphi \wedge T_{2,1}\varphi \rightarrow B_2\varphi$	Axiom C1, PLC believes φ
11.	$w \vDash \mathbb{V}^2_{\varphi}(w) = \text{true}$	$\mathbb{V}^2_{\varphi}(w)$ always returns true

Since $T_{2,1}\varphi \wedge B_2 I_{2,1}\varphi \rightarrow B_2\varphi$, the programmable logic controller believes the lie told in Step 7 in all cases. Therefore, unknown to the entities in SD^2, $\mathbb{V}^2_{\varphi}(w)$ and $\mathbb{V}^2_{\neg\varphi}(w)$ cannot be evaluated. These are the requirements to conclude that φ is MSDND secure from SD^2. □

Theorem 3. *If the system is MSDND secure for SD^2, then any entity i within SD^2, SD^3 and SD^4 will believe all is well.*

Proof. Obviously, $(\varphi \oplus \neg\varphi) = \text{true}$, so the first condition for MSDND is met. If φ cannot be correctly evaluated in SD^2, then both conditions are met. The virus always reports to SD^2 that $\varphi = \text{true}$, so regardless of the status of the cream separator, the infected system reports to SD^2 that all is well. Any entity in SD^2 will report all is well all the way up to SD^4. No matter what, the trusting human will suspect nothing and the cream will be ruined. □

But what if the human is not so trusting? If the human walks over periodically to check the speed gauge on the cream separator, he or she will know instantly that something is wrong and the MSDND security of the virus attack would be broken. Thus, an analysis of the information flows to find MSDND-secure flows could save the ice cream company, but measures must be taken to ensure that all the information flows are not MSDND secure. This leads to the odd result that breaking the security of an attack effectively spoils that attack. Nevertheless, the approach has made it impossible for the attacker to use security against the ice cream company.

Interested readers should note that the modeling and analysis of the cream separator formally shows how the Stuxnet virus operated [15].

6. Conclusions

This chapter has demonstrated the advantages of conducting a formal security analysis of a cyber-physical system that includes the security of the physical assets, the cyber assets and the information flows that are inherent to interdependent cyber-physical systems. Indeed, information flow analysis is most critical because the other aspects of security are well understood.

It is important to examine all the information flows in a cyber-physical system and modify the nondeducible flows to eliminate nondeducibility. This prevents an agent from using nondeducibility against the system. While attacks may not be prevented, all attacks that are not nondeducibility secure can at least be detected early. Additionally, because MSDND is not trace-based like many other information flow security models, it is possible to use MSDND techniques to monitor the security of information flows in real time.

Cyber-physical systems present serious challenges to security professionals. If a cyber-physical system is viewed only as a set of physical things controlled by secure electronics, there is a good chance an adversary could observe the physical actions of the system and deduce the secure actions that must be hidden to protect the cyber-physical system. The complex intertwining or coupling of the two sides of a cyber-physical system produces ample opportunities

for information flows that can be used as attack vectors. Because of the collateral damage that can occur when a cyber-physical system is not properly controlled, an attack could lead to unacceptable property damage and even the loss of human life.

While access control models are inadequate for securing cyber-physical systems, the models help determine the subjects and objects of the cyber side of cyber-physical systems. The Bell-LaPadula model is especially useful for determining the security domains of a system while the Lipner model may help determine the trust roles between subjects. However, access control makes little sense when the security domains of a cyber-physical system are not well defined or overlap in unexpected ways.

Information flow security models are of greatest use in protecting cyber-physical systems. Noninterference and noninference are very useful for discovering the more obvious information flows that can be observed by an adversary, but nondeducibility methods must be used to discover more subtle information flows. MSDND is a very powerful tool for modeling subtle leaks of information that are critical to the operation of cyber-physical systems and can easily be used to find attack vectors that are MSDND secure. A cyber-physical system can then be modified to reduce or eliminate the attacks. This is a necessary step in the race to secure a system before it is attacked.

Acknowledgement

The author wishes to thank the reviewers for their helpful suggestions that have improved this chapter.

References

[1] D. Bell and L. LaPadula, Secure Computer Systems: Mathematical Foundations, Technical Report 2547, Volume 1, MITRE, Bedford, Massachusetts, 1973.

[2] D. Bell and L. LaPadula, Computer Security Model: Unified Exposition and Multics Interpretation, Technical Report ESD-TR-75-306, MTR-2997, Rev. 1, MITRE, Bedford, Massachusetts, 1976.

[3] K. Biba, Integrity Considerations for Secure Computer Systems, Technical Report ESD-TR-76-372, MTR-3153, Rev. 1, MITRE, Bedford, Massachusetts, 1977.

[4] M. Bishop, *Computer Security: Art and Science*, Addison-Wesley, Boston, Massachusetts, 2003.

[5] C. Bryce, J. Banatre and D. LeMetayer, An approach to information security in distributed systems, *Proceedings of the Fifth IEEE Workshop on Future Trends in Distributed Computing Systems*, pp. 384–394, 1995.

[6] J. Butts and S. Shenoi, Preface, in *Critical Infrastructure Protection VI*, J. Butts and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. xv–xvi, 2012.

[7] I. Copi, *Introduction to Logic*, Macmillan, New York, 1972.

[8] T. Fine, J. Haigh, R. O'Brien and D. Toups, Noninterference and unwinding for LOCK, *Proceedings of the Computer Security Foundations Workshop*, pp. 22–28, 1989.

[9] T. Gamage, B. McMillin and T. Roth, Enforcing information flow security properties in cyber-physical systems: A generalized framework based on compensation, *Proceedings of the Thirty-Fourth IEEE Computer Software and Applications Conference Workshops*, pp. 158–163, 2010.

[10] J. Goguen and J. Meseguer, Security policies and security models, *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 11–20, 1982.

[11] J. Goguen and J. Meseguer, Unwinding and inference control, *Proceedings of the IEEE Symposium on Security and Privacy*, p. 75–87, 1984.

[12] M. Harrison, W. Ruzzo and J. Ullman, Protection in operating systems, *Communications of the ACM*, vol. 19(8), pp. 461–471, 1976.

[13] G. Howser and B. McMillin, Modeling and reasoning about the security of drive-by-wire automobile systems, *International Journal of Critical Infrastructure Protection*, vol. 5(3-4), pp. 127–134, 2012.

[14] G. Howser and B. McMillin, A multiple security domain model of a drive-by-wire system, *Proceedings of the Thirty-Seventh IEEE Computer Software and Applications Conference*, pp. 369–374, 2013.

[15] G. Howser and B. McMillin, A modal model of Stuxnet attacks on cyber-physical systems: A matter of trust, *Proceedings of the Eighth International Conference on Software Security and Reliability*, pp. 225–234, 2014.

[16] S. Kripke, A completeness theorem in modal logic, *Journal of Symbolic Logic*, vol. 24(1), pp. 1–14, 1959.

[17] E. Lee, Cyber-physical systems – Are computing foundations adequate? presented at the *NSF Workshop on Cyber-Physical Systems: Research Motivation, Techniques and Roadmap* (`ptolemy.eecs.berkeley.edu/publications/papers/06/CPSPositionPaper`), 2006.

[18] C. Liau, Belief, information acquisition and trust in multi-agent systems – A modal logic formulation, *Artificial Intelligence*, vol. 149(1), pp. 31–60, 2003.

[19] S. Lipner, Non-discretionary controls for commercial applications, *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 2–10, 1982.

[20] S. McCamant and M. Ernst, Quantitative information flow as network flow capacity, *ACM SIGPLAN Notices*, vol. 43(6), pp. 193–205, 2008.

[21] J. McLean, Security models and information flow, *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 180–187, 1990.

[22] A. Myers and B. Liskov, Protecting privacy in a decentralized environment, *Proceedings of the DARPA Information Survivability Conference and Exposition*, vol. 1, pp. 266–277, 2000.

[23] N. Nagatou and T. Watanabe, Run-time detection of covert channels, *Proceedings of the Seventh International Conference on Availability, Reliability and Security*, pp. 577–584, 2006.

[24] C. O'Halloran, A calculus of information flow, *Proceedings of the First European Symposium on Research in Computer Security*, pp. 147–159, 1990.

[25] K. Poulsen, Hacker disables more than 100 cars remotely, *Wired*, March 17, 2010.

[26] A. Sabelfeld and A. Myers, Language-based information-flow security, *IEEE Journal on Selected Areas in Communications*, vol. 21(1), pp. 5–19, 2003.

[27] D. Sutherland, A model of information, *Proceedings of the Ninth National Computer Security Conference*, pp. 175–183, 1986.

[28] U.S. Department of Homeland Security, Cyber Security Overview, Washington, DC (`www.dhs.gov/cybersecurity-overview`), 2015.

[29] J. Wittbold and D. Johnson, Information flow in nondeterministic systems, *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 144–161, 1990.

[30] C. Woodyard, Start, unlock or honk horn of your GM car from a cellphone, *USA Today*, July 22, 2010.