International Journal of Security and Its Applications
Vol. 10, No. 8 (2016), pp.239-250
http://dx.doi.org/10.14257/ijsia.2016.10.8.21

Flow-based Physical Security

Sabah Al-Fedaghi

Computer Engineering Department, Kuwait University, Kuwait
sabah@alfedaghi.com

Abstract

Physical security is a vital part of any organization's operations. In spite of advancements in new technologies for personnel access control and security development life cycles, the need exists for a pre-design schematic specification that bridges the gap between "natural communication" (e.g., English) and semiformal computing-based diagramming descriptions (e.g., UML). Such a specification can play an important role in facilitating understanding among all stakeholders and as a first step to implementation and development of programming-based diagrams. Most of the reported research in the area of physical security has been driven by practical objectives; the value of these studies is limited because of their static representations based on static conceptions of space. This paper proposes to develop a security system based on the notion of security as a machine. The machine is an abstract apparatus with synchronic order of five states (stages): creation, release, transfer, receipt, and process. The resultant model views a security environment in terms of flows of things and uses this flow to establish a system-based representation. The paper introduces a sample of such a map for two cases: (i) an airport luggage handling process that involves a possibly compromised human-based portion, and (ii) an insider threat scenario in which the attacker is one of the personnel allowed to physically access the premises. The resultant depiction seems suitable for security operations, training, and planning.

Keywords*: physical security, computer security, security as a machine, conceptual modeling*

1. Introduction

Security is an essential component of any organization in which the configuration of security mechanisms includes general features such as restricting access and preventing tampering with assets. Security failures are often attributed to a lack of general understanding of the notion of security as fundamental to any system. In a 2012 study of data breach incidents, it was found that the great majority of successful attacks resulted from an organization's poor understanding of security and poor deployment of security measures [1].

Physical security (including physical "computer and network" security) is a critical part of any organization. The increased complexity of physical infrastructures demands designing of effective physical security [2]. It has been found that a significant number of security incidents exploit some vulnerability of physical security [3]. Physical security covers all the devices, technologies, and materials for perimeter, external, and internal protection, *e.g.*, sensors, closed-circuit television, barriers, lighting, and access controls [4]. It addresses the threats, vulnerabilities, and countermeasures to physically protect an enterprise's resources and information [5].

A recent trend in security is to use top-down model-driven approaches including UML profiles for multifaceted security for cyber-physical systems needs [6]. "[S]ecurity of critical infrastructures… is a multi-faceted problem that requires an integrated approach

ISSN: 1738-9976 IJSIA

taking into account digital (*i.e.*, cyber) security as well as physical security, which is strictly related to system protection against intentional threats of [a] physical nature" [6].

According to Pieters [7], the philosophical basis of security research has received little attention until now and "the scientific endeavor of security [has] been left largely unexamined" (see also [8]). Most of "the existing research literature on information security is driven by practical aspirations" [8-9]. The focus is on such questions as how to develop secure information systems and how to prevent abuse of a system, where the main objectives are confidentiality, integrity, and availability (CIA) [10-12].

Recently, Vuorinen and Tetri [8-9] developed a philosophy of information security that delineates the ontology of information security to provide a theoretical base for understanding how an asset can be secured. Vuorinen and Tetri [9] adopt the concept of a "machine" proposed by philosophers Deleuze and Guattari [13], for whom a machine has a function, which is to produce and interrupt [9]. "A machine may be defined as a system of interruptions or breaks" [10]. It is "a machine of order, a machine that seeks to exclude noise, impurities" [8]. Using machine as metaphor is a well-established concept: man, woman, city, brain, stars, software, nature, human body, organization, finite element method, tree, internal grammar of a language, world, economy, factory … all have used machine as a metaphor. For example, according to Dalio [14], the economy is like a machine. The dominance of a new machine, the computer, has pushed this metaphor further.

The computer progressively captured the attention of scientists in both human and cognitive sciences so much that the machine not only became the mainstream metaphor but a paradigm of scientific investigation, in the sense of a reference frame for the community of researchers in the domain. ([15], referencing [16])

Along the lines of Vuorinen and Tetri [8-9], this paper describes a model of security based on the notion of security as a machine, with the intention of facilitating security design. Here the machine is an abstract apparatus with synchronic order of five states (stages): creation, release, transfer, receipt, and process. The resultant model views the security environment in terms of flows of things and uses this flow to establish a system-based representation. The paper introduces a sample of such a map for two cases: (i) an airport luggage handling process that involves a possibly compromised human-based portion, and (ii) an insider threat in which the attacker is one of the personnel allowed to physically access the premise.

In order to develop this security machine, the next section briefly describes a model called the Flowthing Model, FM, that will be used as a conceptual foundation for the diagrammatic specification of abstract machines. FM has been used in many applications [17-20] and is briefly reviewed in the next section. The example given there is a new contribution.

2. Flowthing Model

The Flowthing Model (FM) was inspired by the many types of flows that exist in diverse fields, including information flows, signal flows, and data flows in communication models. This model is a diagrammatic schema that uses *flowthings* to represent a range of items, for example, electrical, mechanical, chemical and thermal signals, circulating blood, food, concepts, pieces of data, and so on. Flowthings *flow* in an abstract flow machine with basic stages where flowthing can be created, released, transferred, processed, and received (see Figure 1). Hereafter, flowthings may be referred to as *things*.

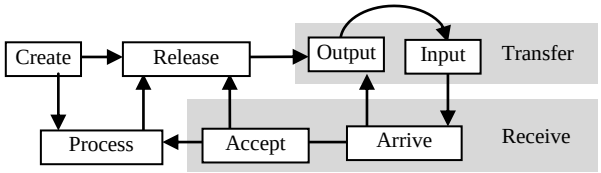


Figure 1. Flow Machine

The machine is the conceptual fiber used to change or transmit flowthings as they pass through stages, from their inception or arrival to their de-creation or transmission. The notion of machine here is close to the idea of "system" or "engine" [21]. Machines form the organizational structure (blueprint) of whatever is described. These processes can be embedded in a network of assemblies called *spheres* in which the processes of flow machines take place. Machine blueprints or maps can be adapted in all types of processes: fire evacuation plans, wayfinding, facility management, security procedures, *etc*.

The stages in Figure 1 can be described as follows:

Arrive: A thing reaches a new machine (sphere).

Accepted: A thing is permitted to enter (*e.g.*, wrong credentials lead to rejection). If arriving things are always accepted, *Arrive* and *Accept* can be combined as a **Received** stage.

Processed (changed): The thing goes through some kind of transformation that changes it without creating a new thing.

Released: A thing is marked as ready to be transferred outside the machine.

Transferred: A thing is transported somewhere from/to outside the machine (sphere).

Created: A new thing is born (created) in a machine.

The machine of Figure 1 is a generalization of the typical *input-process-output* model used in many scientific fields. In general, a flow machine is thought to be an abstract machine that receives, processes, creates, releases, and transfers things. The stages in this machine are mutually exclusive (*i.e.*, a basic thing in the Process stage cannot be in the Create stage or the Release stage at the same time). An additional stage of Storage can also be added to any machine to represent the storage of things; however, storage is not an exclusive stage because there can be *stored processed* flowthings, *stored created* flowthings, *etc*.

FM also uses the notions of *spheres and subspheres*. These are the network environments and relationships of machines and submachines. Multiple machines can exist in a sphere if needed. A sphere can be a person, an organ, an entity (*e.g.*, a company, a customer), a location (a laboratory, a waiting room), a communication medium (a channel, a wire). A flow machine is a subsphere that embodies the flow; it itself has no subspheres. Control of the movement of things is embedded in the stages.

FM also utilizes the notion of *triggering*. Triggering is the activation of a flow, denoted in FM diagrams by a dashed arrow. It is a dependency among flows and parts of flows. A flow is said to be triggered if it is created or activated by another flow (*e.g.*, a flow of electricity triggers a flow of heat), or activated by another point in the flow. Triggering can also be used to initiate events such as starting up a machine (*e.g.*, remote signal to turn on). Multiple machines captured by FM can interact by triggering events related to other machines in those machines' spheres and stages.

Example: Sommerville [22] presented a case study of an insulin pump that simulates the operation of the pancreas by collecting data from a sensor and controlling a pump that

delivers a certain dose of insulin to a user. Some components of the system are explained as follows:

1. Needle receives insulin from pump and delivers it into patient's body.
2. Sensor measures glucose in the blood and sends results to the controller.
3. Pump receives the required amount of insulin dose and pumps it to the needle.
4. Controller

Figure 2 shows a partial view of an activity diagram for this example. Figure 3 shows the corresponding FM representation. The pump system encompasses five (global) spheres: Patient, Sensor, Control, Pump, and Needle. The Patient sphere has a Blood subsphere (circle 1) that in turn has two subspheres, a machine for the blood itself (2) and another for the insulin (3) that is injected into the blood stream. A sample of the blood is taken (Created) and flows to the Sensor (4) to be processed (5), which triggers the generation (6) of data on the level of Sugar. This is sent to the Controller (7), which processes it (8) to trigger a release (9) of the calculated amount of insulin, which is sent to the Pump (10) where it is processed (11) to trigger releasing (12) the corresponding dose to the needle (13). Finally, the needle delivers the dose (14) into the patient's blood (15).

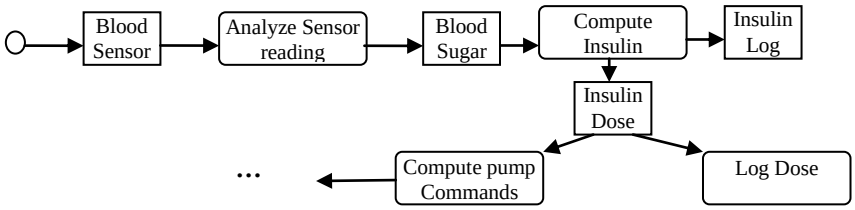


Figure 2. Activity Model of the Insulin Pump (partial, redrawn from [22])

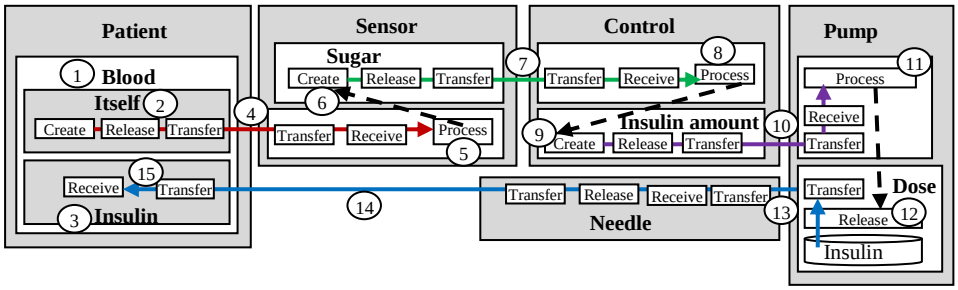


Figure 3. FM Representation of the Insulin Pump

Comparing the activity diagram of Figure 2 with the FM representation of Figure 3, it can be noticed that the structure of the activity diagram is "flat," "spreading" conceptual units across the depiction with infinite activities: analyze, compute, log, and potentially any English verb. Other UML diagrams help in providing more structural aspects in modeling. No matter the final representation of these diagrams, security modeling needs tighter continuity in composition of elements and activities to avoid any gaps and breaks in control of the thrust and rush of events as they unfold. FM exhibits such tightness, a top requirement for security applications.

3. Security as a Machine

For the Internet Engineering Task Force (IETF), "security is a system condition in which system resources are free from unauthorized access and from unauthorized or accidental change, destruction or loss. Safety is the property of a system being free from risk of causing harm (especially physical harm) to its system entities." [23]. According to Nunes-Vaz and Lord [2], "Security is concerned with the risks originating from the environment and potentially impacting the system, whereas safety deals with the risks arising from the system and potentially impacting the environment." In FM terminology, security is a machine with the function of handling security/safety concerns: malfunctions in different types of flows and triggering that result in malicious behavior or effects (see Figure 4). In the words of Deleuze and Guattari [13], "a machine has a function, which is to produce [create] and interrupt [trigger]." Additionally, it transfers, processes, releases and receives. In the security context, these operations are employed for watchdogging.

For example, *fuses* in electrical circuits protect the system from excessive current (external flow) that can cause damage and fire, as illustrated in Figure 5.

Specifically, *the problem examined in this paper* is to *design* a security machine to handle (create, release, transfer, receive, and process) objectionable flows and triggering that cause malicious effects in a system.

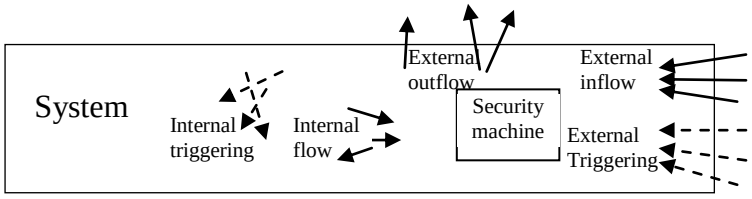


Figure 4. Visualization of Physical Security as a Machine Environment

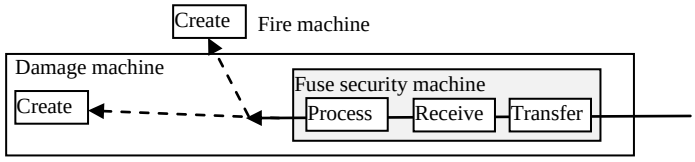


Figure 5. Example: Electrical Circuits Protect the System from Excessive Current

4. Sample Design

To illustrate a type of security design based on FM, consider Nunes-Vaz and Lord's [2] study case of a hypothetical facility required to reduce the risk associated with arrangement of threats to its physical security, visualized as a nested arrangement of zones. The required system is configured to prevent a hypothetical terrorist scenario involving a vehicle-based improvised explosive device, with the attacker also carrying firearms to coerce entry.

Figure 6 shows the initial unpopulated stage of the design. "The three function containers are initially unpopulated and the inevitable failure of the (unpopulated) detect function (shown by a dotted arrow below the detect container) leads to the least desirable output state of 'inside without alarm'" [2]. Figure 7 shows a refinement of Detect.

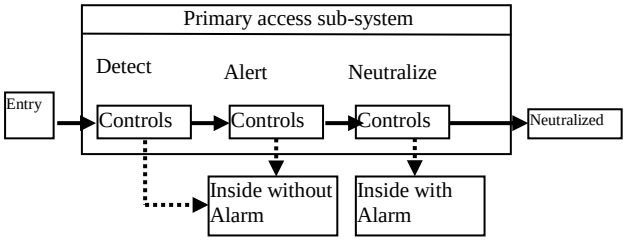


Figure 6. Visualization of Initial Design of Physical Security (Partial, Redrawn from [2])

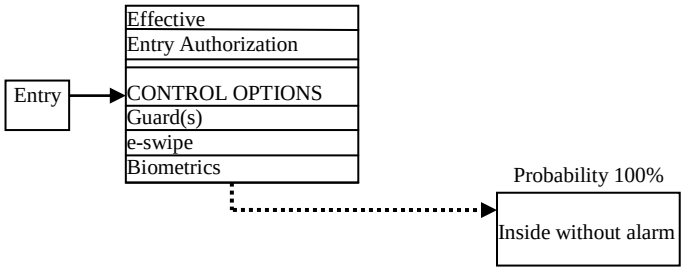


Figure 7. Refinement of the Detect Function (Partial, Redrawn from [2])

In the next subsection, two cases of FM-based security are introduced. However, before leaving Nunes-Vaz and Lord's [2] model, we describe it in FM language to illustrate the *conceptual* flows used in FM.

A possible visualization of Nunes-Vaz and Lord's [2] model is shown in Figure 8. Detection, Alarm, and Neutralization are types of spheres that can be described by their flows. The flowthings (persons) enter the detection area (circle 1) and leave (2). If there is a positive detection, it triggers (3) the creation of an alarm, which in turn triggers transferring the cause (4) to be neutralized (5). Note how the *flow* in the figure "moves" from the state of Detection to the state of Neutralization, even though it may be that both detection and neutralization occur in the *same physical place*.

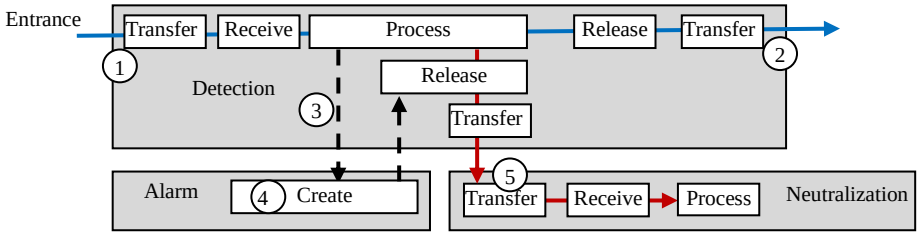


Figure 8. FM Description of the Detect, Alarm, and Neutralize "functions"

4.1 Case Study: Airport Handling Process

Recent news that a terrorist smuggled a bomb into an airplane presents a suitable study case for designing a practical security system. In this context, "A security team is looking at the passenger-screening process and baggage handling procedures, and assessing the *effectiveness* of background checks on airport staff" [24; italics added].

In the case of the terrorist act mentioned above, it seems that the compromised area was the luggage compartment of the airplane because, when tourists were asked to return

home, airlines refused to load their luggage, which had been shipped separately on different planes. Assuming that the bomb was smuggled through the luggage handling system with a possible accomplice on the handling team, how can the security system be designed to make such an event highly unlikely in the future?

Clearly, success in smuggling a bomb onto a plane is a process failure that raises the issue of the nature of a security system of the type under discussion (*e.g.*, involving a human factor). We can observe that such a system includes a nonsystematic process (*e.g.*, as in the weather prediction process). In such a process "there is no … set of insights from which the whole process and all its events may be mastered. The difficulty of investigating them evidently increases with the number of diversity of the separate intelligibilities involved" [25]. Nonsystematic processes involve application of probabilities and statistics to calculation of risk, *e.g.*, the likelihood that an attempted physical attack is successful. In the case under study, the probability of execution can be based on such things as religion, location, culture, *etc*., along with continuously updated statistical data. One alternative method for development of a process with a lower likelihood of events is that of physical *redundancy*.

Redundancy is one of the options for ensuring a higher level of protection against such vulnerabilities that result in catastrophic events. "Redundancy as a general approach is clearly understood to be a valid defense against physical faults. There is a rich set of understood design 'tricks' that use redundancy… in terms of tradeoffs between cost of the redundancy during normal operation and the effectiveness and performance degradation in case of failure" [26]. It provides a multilayer defense effective in preventing single points of failure.

FM is a very suitable tool for modeling a security system of this type because of its structural properties based on flows. Upon these flows is laid the "territory of events" to be regulated with redundant crisscrossing of multiple controls, as follows:

- Redundant multiple automatic inspection of a piece of luggage, *i.e.*, by different machines
- Redundant multiple inspection of luggage handlers, *i.e.*, by independent teams

Consider the case of multiplicity in two independent handling systems (*e.g.*, for VIPs, for whom it could be even more). Such a two-level security process is not an uncommon situation; some airlines and agencies have their own system of checking persons and bags in addition to the airport systems. Figure 9 shows the FM scenario for two teams: blue and green, under such conditions. A member of the blue team enters the blue team gate (circle 1) where he/she is processed according to his team security procedure (2), then the blue team member enters the green team inspecting area where the person is processed again (3) according to the green team procedure. He/she is then sent back to the blue team area (4) to be allowed to enter the blue team luggage area to do his/her work of searching luggage (5). A similar flow is designed for the green team members. Thus, each team member is checked by the procedures of his own team and also by those of the other team. A piece of luggage flows to the green team luggage area (6) where it is processed (7) and then flows to the blue team luggage area (8) where it is processed according to blue team procedure (9).

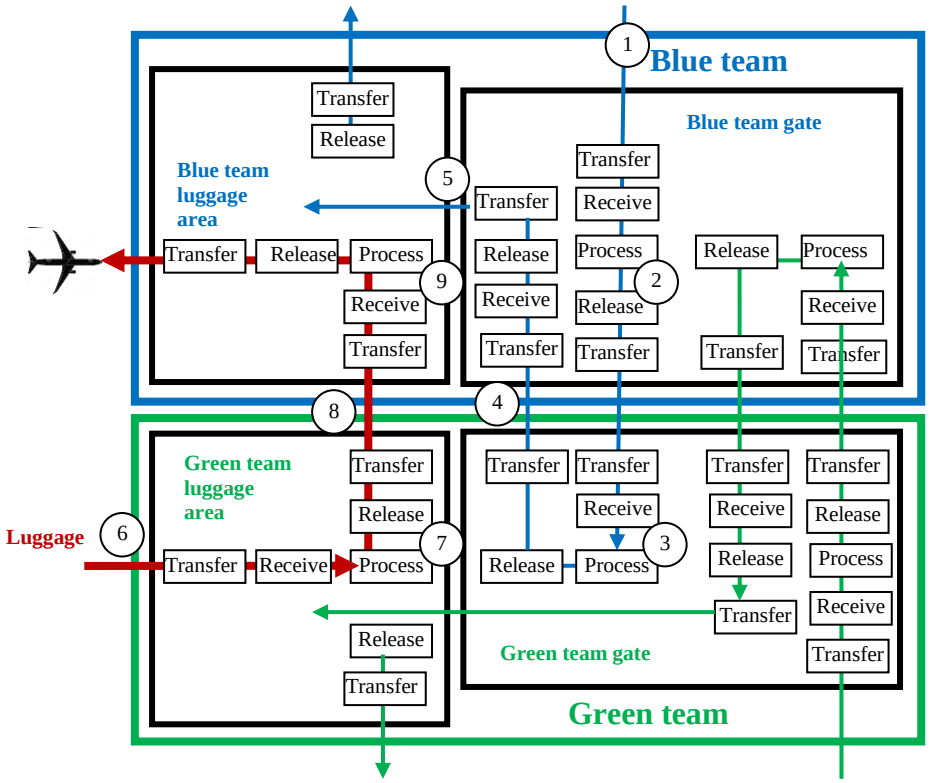


Figure 9. FM Description of a Multiple Process System for Two Teams of Luggage Handlers

Now, assuming effective security procedures, the system breaks down when:

- Three accomplices, two from the blue team and one from the green team, conspire to work at the same time, when one of them passes his/her own team's checking procedure and the procedure of the other team to smuggle explosives and place them with the flowing luggage. In this case, it is assumed that the blue team conspirator ignores his/her procedure and that he/she manages to move from checking persons to checking luggage.

- Two members from the two teams, working the same time shift, ignore explosives passed as luggage, and ignore the results of the two processes.

4.2. Case study: Lineside Shelter Scenario

Marrone *et. al.*, [6] describe a lineside shelter protection system where shelters are small buildings located along railway tracksides. The shelter is accessed by a door controlled by an Access control device (*e.g.*, a card reader). The entrance room accessible by the door features intrusion presence detection sensors and a fog generator to temporarily blind intruders. A rack in this room contains two servers: a Traffic Server that controls the railway devices and a Cyber-Physical Protection (CPP) Server that is responsible for monitoring the shelter's security (see Figure 10). Figure 11 depicts a UML Deployment diagram describing the "cyber-physical architecture" of the shelter. Figure 12 depicts the annotated UML Sequence Diagram illustrating the behavior of an Insider Threat scenario. In the annotated UML diagrams, a transformation generates yet another diagramming-based model suitable for evaluating the success probability of an attack [6].

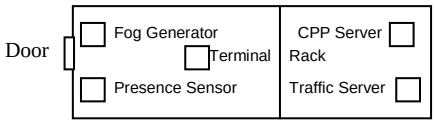


Figure 10. Redrawn Partial View of Lineside Shelter Reference Plant [6]

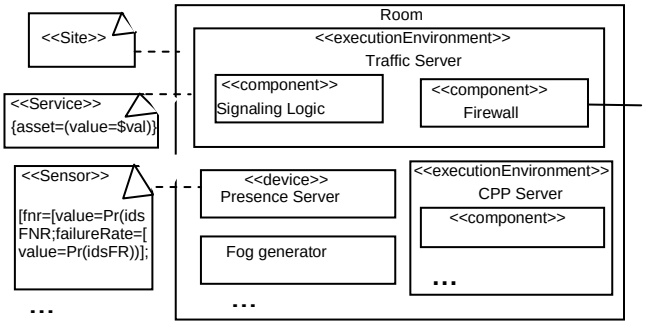


Figure 11. Redrawn Partial View of UML Deployment Diagram of the Shelter [6]

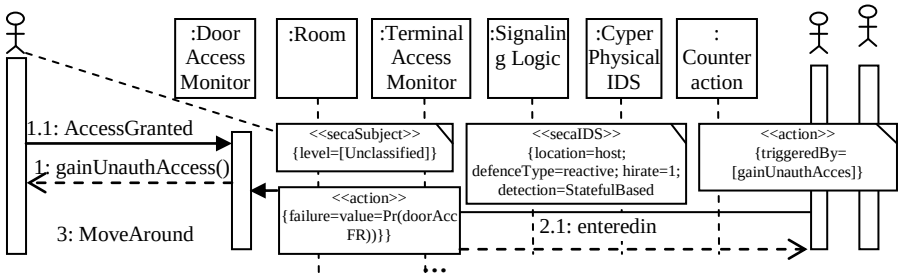


Figure 12. Redrawn Partial View of the Annotated UML-SD of the Insider Threat Scenario [6]

It can be observed that whatever the advantages of these diagrams (*e.g.*, machine readability) their composition is loose for a conceptual foundation for modeling security, because descriptive gaps can appear throughout the design. Also, for someone who is not a software engineer (*e.g.*, stakeholders), the description looks like gibberish, a diagrammatic representation that cannot easily be understood or used in reasonable communication and discussion about security matters. The diagrams may be needed as pre-(model)programming diagrams in a role similar to the role of flowcharts in a pre-specification step for programming. At the level between "natural communication" (*e.g.*, spoken language) and such semiformal specification there is a need for a language that provides a limited notation with an underlying event-tightened model that is easily understood.

Consider an insider threat in which the attacker is part of the personnel allowed to physically access the shelter room but not allowed to log into the computer system with high-privilege credentials. In the case study, by correlating physical and logical credentials, the CPP Server recognizes the intrusion and activates cyber-countermeasures such as notification to the control room [6]. Figure 13 shows the FM representation of this scenario. First, the attacker inserts his/her card (circle 1), which triggers (2) creating data (3) that flow (4) to the CPP Server to be processed (5) to trigger (6) opening the door (7). The attacker then enters the room (8) and attempts to login (9), which is detected by the server (10). Accordingly, a notification is created and sent to the control room (11). Note that the FM stages embed control procedures. The figure can be enhanced using the same methodology with other spheres such as communication with the network and control spheres of the presence sensor and fog generator. Additionally, the description can be further detailed depth-wise, *e.g.*, the login sphere (10) is viewed as a software module with subspheres dealing with access permissions, a threat evaluation module, *etc*.

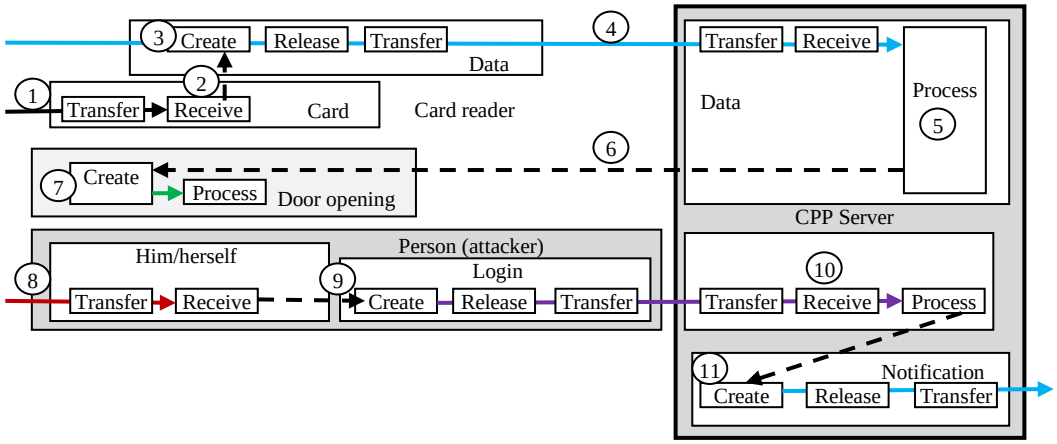


Figure 13. FM of Basic Insider Threat

For example, Figure 14 shows the FM description of an attacker with permission to login (12). Thus, a session is opened (13) in which he/she issues commands (14) that are transmitted to the control unit (15), which rejects implementing them (16). Accordingly, the session and the login are terminated (17 and 18, respectively). Note that, for simplicity's sake, the session stages are drawn without a box and Create and De-create are represented in the same stage (internal actions in the Create stage).
The point here is that such a diagrammatic representation presents a uniform view to describe the threat scenario and needs little annotation to facilitate understanding by all stakeholders and as a first step to implementation and development of programming-based diagrams such as the ones described previously.

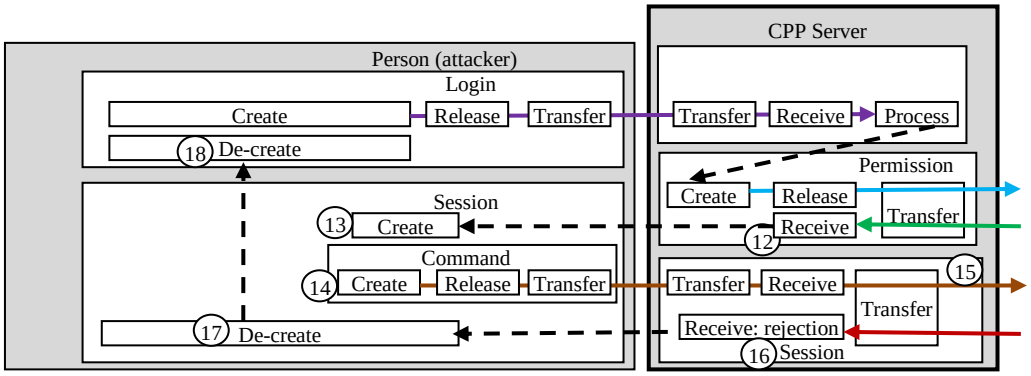


Figure 14. Insider Threat with Permission to Login

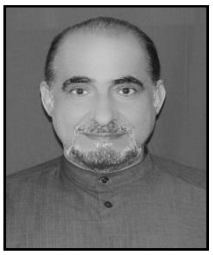
5. Conclusion

This paper has proposed to develop a depiction of security processes based on the notion of a security apparatus as a machine. The resultant model views a security environment in terms of flows of things and uses this system of flows to establish a system-based representation. The resultant description exhibits tightness (no gaps), the topmost requirement in security applications. This tightness is provided by continuity in composition of elements and activities to avoid any breaks in control of events. The method is applied to two study cases in the area of physical security. The result seems promising as a vehicle for modeling attacks and a security plan. Further research would use FM in more complicated design cases.

References

[1] S. Schiavone, L. Garg and K. Summers, "Ontology of Information Security in Enterprises", Electronic Journal of Information Systems Evaluation, vol. 17, no. 1, **(2014)**, pp. 71–87.
[2] R. Nunes-Vaz and S. Lord, "Designing Physical Security for Complex Infrastructures", International Journal of Critical Infrastructure Protection, vol. 7, no. 3, **(2014)**, pp. 178–192.
[3] B. Spirovski, "Datacenter Physical Security Blueprint", Information Security Short Takes (2015 – Access). http://www.shortinfosec.net/2008/04/datacenter-physical-security-blueprint.html, **(2008)**.
[4] S. Niles, (**no date**) "Physical Security in Mission Critical Facilities", Schneider Electric White Paper 82, Revision 2 (2015 – Access). http://www.apcmedia.com/salestools/SADE-5TNRPL/SADE-5TNRPL_R2_EN.pdf
[5] S. L. Ksander, "ITNS and CERIAS CISSP Luncheon Series: Physical (Environmental) Security", Datashow presentation. (2015 – Access). http://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=13&cad=rja&uact=8&ved=0ahUKEwi4j5Lxs5_JAhULWRoKHWnQBBA4ChAWCCQwAg&url=http%3A%2F%2Fwww.purdue.edu%2Fsecurepurdue%2Fdocs%2Ftraining%2Fphysicalsecurityslk.ppt&usg=AFQjCNHoN1In7Oh9XRAS3krw1hrT6LkbKA&sig2=Ip7npKOjQPcIJ0a7RWEDSg
[6] S. Marrone, R. J. Rodríguez, R. Nardone, F. Flammini and V. Vittorini, "On Synergies of Cyber and Physical Security Modelling in Vulnerability Assessment of Railway Systems", Computers and Electrical Engineering, Available online August 2015. DOI: 10.1016/j.compeleceng, **(2015)** August 11.
[7] W. Pieters, "The (Social) Construction of Information Security", The Information Society, vol. 27, no. 5, **(2011)**, pp. 326–335.
[8] J. Vuorinen and P Tetri, "The Order Machine: The Ontology of Information Security", Journal of the Association for Information Systems, vol. 13, no. 9, **(2012)**, pp. 695-713.
[9] J. Vuorinen and P. Tetri, "Security as a Machine: Struggling between Order and Chaos", in Pacific Asia Conference on Information Systems (PACIS) 2009 Proceedings. Paper 113, **(2009)**. http://aisel.aisnet.org/pacis2009/113
[10] G. Dhillon and J. Backhouse, "Current Directions in IS Security Research: Toward Socio-Organizational Perspectives", Information Systems Journal, vol. 11, no. 2, **(2001)**, pp. 127–153.
[11] G. Dhillon, "Principles of Information Systems Security: Texts and Cases", John Wiley & Sons, NJ, **(2007)**.
[12] J. Stanton and K. R. Stam, "The Visible Employee: Using Workplace Monitoring and Surveillance to Protect Information Assets – without Compromising Employee Privacy or Trust", Information Today, NJ, **(2006)**.
[13] G. Deleuze and F. Guattari, "Capitalism and Schizophrenia", vol. 1, "Anti-Oedipus", Continuum, London, **(2004)**.
[14] R. Dalio, "Man and Machine," The Economist, Mar. 10, **(2012)**. Draft version. http://www.economist.com/node/21549968.
[15] Kohler, "To Think Human out of the Machine Paradigm: Homo ex Machina", Integrative Psychological and Behavioral Science, vol. 44, no. 1, **(2010)**, pp. 39-57.
[16] T. S. Kuhn, "The Structure of Scientific Revolutions", University of Chicago Press, Chicago, **(1962)**.
[17] S. Al-Fedaghi, "Heraclitean Ontology for Specifying Systems", International Review on Computers and Software (IRECOS), vol. 10, no. 6, **(2015)**. http://dx.doi.org/10.15866/irecos.v10i6.6493
[18] S. Al-Fedaghi and N. Aljallal, "Conceptual Schematization of Microcontroller and Assembly Language", International Journal of Software Engineering and Its Applications, vol. 8, no. 10, **(2014)**, pp. 179-190.
[19] S. Al-Fedaghi, "Schematizing Proofs based on Flow of Truth Values in Logic", presented at IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC 2013), Manchester, UK, **(2013)**, pp. 194 - 200.
[20] S. Al-Fedaghi and F. Al-Kanderi, "Integrating Security Concerns into Software Development", International Journal of Security and Its Applications, vol. 7, no. 3, **(2013)**, pp. 235-248.
[21] H.-J. Rheinberger, "Toward a History of Epistemic Things: Synthesizing Proteins in the Test Tube", Stanford University Press, Stanford, CA, **(1997)**.
[22] Sommerville, "Software Engineering", 9th ed., **(2011)**.
[23] R. Shirey, "Internet Security Glossary", version 2, Internet Engineering Task Force (IETF), RFC 4949, **(2007)** Aug.
[24] S. Calder, "Russian Plane Crash Q&A: Why Has Russia Now Confirmed Metrojet Flight 9268 Was Bombed?" Independent, November 17, 2015, http://www.independent.co.uk/news/world/europe/russian-plane-crash-qa-why-has-russia-now-confirmed-metrojet-flight-9268-was-bombed-a6737701.html
[25] H. A. Meynell, "An Introduction to the Philosophy of Bernard Lonergan", University of Toronto Press, **(1991)**.
[26] B. Littlewood and L. Strigini, "Redundancy and Diversity in Security", In Computer Security – ESORICS, Vol. 3193 of the series Lecture Notes in Computer Science, Springer, **(2004)**, pp. 423-438.

Author



Sabah Al-Fedaghi, He holds an MS and a PhD in computer science from the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, Illinois, and a BS in Engineering Science from Arizona State University, Tempe. He has published two books and more than 220 papers in journals and conferences on software engineering, database systems, information systems, computer/ information privacy, security and assurance, information warfare, and conceptual modeling. He is an associate professor in the Computer Engineering Department, Kuwait University. He previously worked as a programmer at the Kuwait Oil Company, where he also headed the Electrical and Computer Engineering Department (1991–1994) and the Computer Engineering Department (2000–2007).