On Cybersecurity Science and Engineering*

Francesco Beltramini[1] and Marco Rocchetto[1]

[1] *V-Research, Verona, Italy*

Abstract

The objective of this research is to develop of a theory that defines (all and only) the possible insecurity and security configurations of any abstract system. The theory is structured upon other theories that defines how a component of a system can be abstracted into an agent, defining how agents can be formalized (both syntactically and semantically) to describe an abstract system, such as a graph. Some of these theories (e.g. used for the semantic definition of the abstract system) are the epistemological definition of knowledge, the Belief-Desire-Intent and the Assertion-Belief-Fact framework of reference, mereology, and topological structure. We argue that a mereology is the most appropriate abstract underlying structure, due to its generality, for defining the expressiveness of the system abstraction. Furthermore, a mereology allows us to define an ontology rather than a taxonomy. We also correlate different abstractions of the system to the TRL and the engineering V-model.

We implemented a formal theory (of axioms) of a mereotopology, and of the Region Connection Calculus (RCC3 and RCC5) in a Python program that uses the Z3 SMT solver. The results show that a single component (i.e. agent) of an abstract system has a definite number of different insecurity configurations (e.g. 53 using RCC5 over a topological structure) and only 1 secure (i.e. expected) configurations. The configurations are reported as models satisfying the abstract system semantics.

We considered the philosophical definition of truth behind our approach, rejecting "proof" by induction from partial empirical evidences. Our theory can be applied to system engineering and we show a concrete application of our theory to the risk assessment of an ad-hoc system. Finally, we provide a number of ideas to support the engineering of secure systems (e.g. purely cyber or cyber-physical).

*Confidential, restricted to the authors. Property of V-Research

1 Introduction

Humanum est errare

Seneca the Elder

The European Commission states in[24] that: “Cybersecurity is one of the priority areas [. . .] of the Commission initiative on ICT Standards, which is part of the Digitising European Industry[25] strategy launched on 19 April 2016. The aim is to identify the essential ICT standards and present measures to accelerate their development in support of digital innovations across the economy”. The same document (i.e.[24]) states that “The EU will invest up to €450 million [. . .], under its research and innovation programme Horizon 2020”. The EU, in 2016 published a press release[26] in which they present a strategy to invest €1.8 *billion* to “increase measures to address cyber threats”. The EU is not the only investor in cybersecurity, most of the developed countries and several companies are investing enormous amount of money towards various aspects of cybersecurity (e.g. The US vulnerability databases[77] maintained by the National Institute of Standards and Technologies, i.e. NIST, of the US Department of Commerce).

The cybersecurity industry is growing fast, e.g. as reported in[22]. For example, in[85], published by the Forbes, is stated that €5.3 billion of funding where poured by venture capitalist into cybersecurity companies in 2018. The Forbes, in the same article, also highlights another peculiar (as seemly contradictory) trend: “[. . .] during the same time period, the number of cybersecurity breaches increased exponentially”. The data reported by the NIST through the official CPE (Common Platform Enumeration) Dictionary Statistics on the NVD websites in [78], show that in 2016 the number of reported vulnerabilities reported where around 6000 while in 2019 the number of vulnerabilities was above 16000. The scientific community also reports seemingly unjustifiable findings. In fact, in[38], Cormac Herley (Microsoft Research) shows how basic cybersecurity principles (such as the confidentiality benefit over the clear text for passwords typed into forms, e.g. for logins in websites) are not fully understood or shared between the cybersecurity research community[58]. The lack of understanding of basic security principle, the inverse proportionality between investments in cybersecurity and the number of reported vulnerabilities year after year, can be linked to the lack of a foundational theory on cybersecurity, as already highlighted by Cormac Herley in[40].

Another way of looking at the same problem is by analyzing the different definitions of security with respect to the scientific method of enquiry applied to get to the definition itself. In the following we categorize the related work into three categories, as defined by Sextus Empiricus in [29]. To the best of our knowledge, there is no evidence that this categorization isn’t complete. “The natural result of any investigation is that the investigators either discover the object of search or deny that it is discoverable and confess it to be inapprehensible or persist in their search. [. . .] This is probably why”[29]:

- The *dogmatists* “have claimed to have discovered the truth” on what cybersecurity is
    - Wikipedia defines cybersecurity in [48] as the protection of computer systems and networks from the theft of or damage to their hardware, software, or electronic data, as well as from the disruption or misdirection of the services they provide
- The *academics* “have asserted that it cannot be apprehended”
    - Eugene H. Spafford, Professor at Purdue University, defines cybersecurity as follow. “The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards — and even then I have my doubts.” [73]
- The *skeptics* “go on inquiring”
    - Cormac Herley reaches the conclusion that cybersecurity has no definition. “There is an inherent asymmetry in computer security: things can be declared insecure by observation, but not the reverse. There is no test that allows us to declare an arbitrary system or technique secure. This implies that claims of necessary conditions for security are unfalsifiable. ” [40]

The method applied in our line of research follows what the scientific method mandate. We start by detailing the problem statement by reporting the state of the art in Section 2, we formulate (inductively)[1] hypothesis based on our review (Section 3) which we use to propose a theory in Section 4 that apply to a abstract and standard (i.e. based on standards) secure process development lifecycle based on requirement engineering in Section 5 (with an ad-hoc running example, based on the SWaT testbed [53]). We propose a tool for the risk assessment of a design based on our theory in Section 6. Finally, in Section 7 we conclude the paper.

2 Problem Statement

In[40], Cormac Herley explores what he calls “an asymmetry in computer security”, which he defines as follows: “Things can be declared insecure by observation, but not the reverse. There is no observation that allows us to declare an arbitrary system or technique secure”. Herley then uses this argument to show that “claims that any measure is necessary for security are empirically

[1] “So, whenever they argue “Every man is an animal and Socrates is a man; therefore Socrates is an animal,” proposing to deduce from the universal proposition “every man is an animal” the particular proposition “Socrates therefore is an animal,” which in fact goes (as we have mentioned) to enstablish by way of induction the universal proposition, the fall into the error of circular reasoning, since they are enstablishing the universal proposition inductively by means of each of the particulars and deducing the particular proposition from the universal syllogistically.” Sextus Empiricus, Outlines of Pyrrhonism II-195 [29]

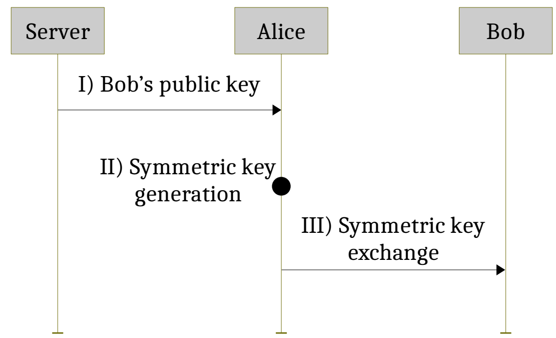


Figure 1: Abstraction of an ad-hoc esemplificative protocol execution

unfalsifiable". Given that, any theory which is not falsifiable by an empirical experiment is well known[2] to be nonscientific (i.e. unfalsifiability is a fallacy of a theory), Herley concludes that there is no scientific theory on cybersecurity; which means that cybersecurity lays in the realm of pseudo-sciences[39]. Herley, e.g. in[37], discusses the implications of a nonscientific approach to cybersecurity, and highlights the tremendous impact on all the scientific research and engineering of systems; leading often to terrorism and wars, and wasting of resources in useless protections or overspending. While the criticism is investigated in[40], no solution is provided. On the contrary, the goal of this work *is* to lay the foundations of a scientific cybersecurity theory. Furthermore, in Section 2.1, we consider the problem raised by Herley not confined to "computer security" but to any abstract system (so that our theory may hold for any sound implementation such as networks, mechanical, cyber, or cyber-physical system, or even a single computer or a single device such as an hard-drive). There is also an apparent inconsistency in[40] that we seek to clarify before following (as we agree) the scientific path draw by Herley: cybersecurity is defined as an abstract property in many formal approaches to the investigation of the security of systems, and the security of the design of a formally verified protocol is indeed falsifiable. For example, in the protocol verification community, security is often defined as a formalization of the high-level properties confidentiality, integrity, and availability. The problem in such approaches is not the definition of what cybersecurity is, but the use of theories (such as the Dolev-Yao attacker model[3][27]) that only applies to specific instances (often called scenarios) and abstraction of the protocol. This, in turn, creates a false sense of security since requires assumptions on the abstraction of the system of which security is verified. As an example, for the formal security verification of the system in Figure 1, a formalized scenario needs to be defined by a modeler

[2] "A theory which is not refutable by any conceivable event is nonscientific. Irrefutability is not a virtue of a theory (as people often think) but a vice." – Karl Popper, Conjectures and Refutations[61]

[3] For the sake of simplicity, the Dolev-Yao attacker can be considered as an abstraction of an active attacker who controls the network but cannot break cryptography.

who chooses (among others): (i) a scope of the formalization (e.g. excluding the server that distributes the public key is often done when verifying the security of authentication protocols), (ii) the number of sessions (even tough some approaches do reason on an infinite number of sessions such as[31]), (iii) honesty/dishonesty of the peers (e.g. in the ASLan++ language[86]), and (iv) the abstraction of the cryptographic primitives (e.g. ProVerif vs CryptoVerif[6]). Some of the choices will completely change the results of the formal verification of the system. For example, under the perfect cryptography assumption[4] and assuming that no violation to any security property is done after message I); in Figure 1, the freedom of choosing the scope determines that the flaws related to the dishonest impersonation of the Server may or may not be considered in the verification process. This choice has tremendous impact on the focus and findings of the verification of the security of the protocol. While this may seem to turn upon minutiae and foreseeable, this highlights the false sense of security that may derive from a non-scientific theory of system security[5].

2.1 Sicurezza: Safety and Security

In most of the natural languages, and in Italian too, the concepts of safety and security are not syntactically differentiated and both terms (safety and security) are expressed by the same word, e.g. sicurezza in Italian. A semantic distinction between safety and security is correlated to a belief[6] that safety deals with *accidents* (i.e. an unfortunate incident) posed by the natural environment (e.g. natural events such as wearing of hardware components) while security deals with *incidents* posed by mankind (e.g. attackers and bugs). The fundamental difference between nature and mankind (and, in turn, between safety and cybersecurity) is believed to be on the different intents[7] (accidents are unfortunate while incidents are not) of the causes that generates the threat; namely, nature is believed not to have malicious intents (but unfortunate causes-effects) while threats generated by mankind are believed to be malicious[8]. An overview on

[4]As defined in[66]: "In the so called perfect cryptography assumption, the security encryption scheme is suppose to be perfect, without any exploitable flaw, and so the only way for the attacker to decrypt a message is by using the proper key. That assumption is widely accepted in the security protocol community, and most of the formal reasoning tools for the analysis of security protocols abstract away the mathematical and implementation details of the encryption scheme[83, 4, 3, 67]"

[5]"To the superficial observer, the analysis of these forms seems to turn upon minutiae. It does in fact deal with minutiae, but they are of the same order as those dealt with in microscopic anatomy." – Karl Marx, Capital Volume 1, 1867

[6]A belief has to be intended as a proposition which is supposed to be true by the majority of humans in our society without a scientific underlying theory but based on partial empirical evidences or inductive proofs.

[7]"The belief–desire–intention software model (BDI) is a software model developed for programming intelligent agents."[46]. In the BDI model, the intents represents the deliberative state of an agent which determines the choice of that agent on what to do.

[8]Of course, logical flaws or bugs may be introduced by other means (e.g. ignorance) without explicit malicious intents, but the exploitation of those flaws is considered (for now, and detailed afterwards in the article) malicious, and then we consider any vulnerability to be malicious (without loss of generality) even if due to the lack of skills.

the aforementioned aspects of safety and security is depicted in Figure 2 and is used as a baseline for a definition of the terms that structure our current understanding of safety and security.

- *Mankind* "refers collectively to humans" [50], while the concept of *Nature* is related "to the intrinsic characteristics that plants, animals, and other features of the world develop of their own accord" (e.g. the physical universe)[51].
  - So far, we have used several terms to refer to an *attacker*, i.e. threat agent or threat source, considering those terms to be semantically equivalent. This "shallowness" raise form the necessity of properly citing the different sources, but, in the reminder of this paper, we consider the Causality principle to be the *threat source*, Nature or Mankind to be the *threat agents* and an *attacker* as a specific malicious threat agent which materialize a threat.
- *Vulnerability*[9], as defined in[57] (and adopted in[5]), is "weakness in an information system, system security procedures, internal controls, or implementation that could be exploited by a threat source". On the one hand, the definition is broad to enclose as much causes (that generates a vulnerability) as possible; on the other hand, it derives from empirical evidences (which should be considered beliefs[10] since they are partial results in nature) while a vulnerability should be defined in a way that is empirically falsifiable. This means that the term vulnerability should have a complete and sound definition, so that no other causes (e.g. other sources) but the ones in the definition are responsible for a vulnerability. Furthermore, the term "threat sources" used in the definition in[57] may be identified with both Nature and Mankind, not differentiating between safety and security. In Definition **??**, we provide a formal theory of vulnerability (so that the scientific community can identify tests for the completeness and soundness of the definition itself).

Most of the safety-preserving principles in the field of engineering of safety-critical cyber-physical systems (such as elevators and aircraft), upon which safety requirements (e.g. in standards such as the IEC 61508 or 61511[1]) are defined, have been defined following empirical tests and measurements. While reasoning by induction based on the empirical observation should be avoided, since it may easily lead to false beliefs instead of scientific theories, this approach is often justified by the supposed impossibility of defining a theory that

[9] The term vulnerability is not present in the Encyclopedia of Cryptography and Security, while it is used in 12 entries (such as in the definition of "penetration testing"[7]) highlighting how commonly this word is used without a proper supporting semantics

[10] "For this view, that *That Which Is Not* exists, can never predominate. You must debar your thought from this way of search, nor let ordinary experience in its variety force you along this way, (namely, that of allowing) the eye, sightless as it is, and the ear, full of sound, and the tongue, to rule; but (you must) judge by means of the Reason (Logos) the much-contested proof which is expounded by me." – Parmenides of Elea, On Nature (circa 500 B.C.), fragments B7.1–8.2 [36]

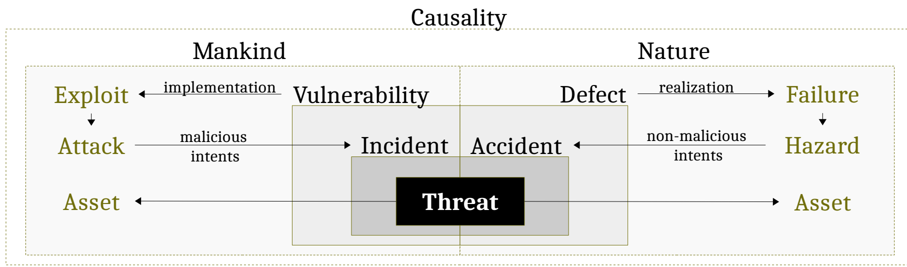


Figure 2: Overview security and safety keywords

correctly predicts failures which, in turn, pose hazards to a system. To the best of our knowledge, and supported by[40], the correlation between predictability of environment and believed unpredictability of attackers (i.e. a malicious environment) has not been correlated to a theory on cybersecurity. Therefore, inductive research efforts in predicting malicious effects are accepted (and published) in scientific conferences (e.g. [65]). A failure of a wire due to environment (e.g. due to humidity, dust, heat &c) is defined from empirical evidences and processes have been standardized to test qualities of hardware components This process completely breaks down when a malicious environment (i.e. an attacker) is considered instead of the (supposedly honest and predictable) natural environment. Therefore, the same approach that is in use for safety, seems not to be applicable to test security.

Going back to Figure 2, a vulnerability does not necessarily become a threat for the system, unless exploited "through a channel that allows the violation of the security policy [. . .]"[57] (e.g. a software or procedure) that takes advantage of the vulnerability causing an *attack* to the system, which may result in several correlated incidents and threats. The process of exploitation of a defect as a vulnerability is reported in Figure 2 such that the difference between exploit and failure, and attack and accident is to be found just in the maliciousness of the intents that causes this process (i.e. excluding the intent, the terms are just syntactic transformation from a vulnerability to defect, from accident to incident). In the following, we conclude the informal definition of the terms that we used in this section and in Figure 2.

- *Causality* refers to the causality principle; defined in[23] as "Causality is a genetic connection of phenomena through which one thing (the cause) under certain conditions gives rise to, causes something else (the effect). The essence of causality is the generation and determination of one phenomenon by another. In this respect causality differs from various other kinds of connection, for example, the simple temporal sequence of phenomena, of the regularities of accompanying processes".

- An *Exploit*[11] is "An exploit (from the English verb to exploit, meaning to

[11] We note that the term exploit is only used as a verb in[76]

use something to one's own advantage) is a piece of software, a chunk of data, or a sequence of commands that takes advantage of a bug or vulnerability to cause unintended or unanticipated behavior to occur on computer software, hardware, or something electronic (usually computerized)."[49].

- An *Attack*, as defined by the International Standard ISO/IEC 27000 is an "attempt to destroy, expose, alter, disable, steal or gain unauthorized access to or make unauthorized use of an asset"; where an *Asset* is "anything that has value to the organization". We note that for the purpose of this article, we do not want to focus on a specific organization or business to define asset but, in general, on any abstract organization (e.g. a company or a society). We do not consider ethical hackers as attacking a system. In fact, we consider the term *hack* as non-malicious (see Hacker[74]).
- A *Threat*, as defined in[57], is "Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, or the Nation through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service".
- *Defect*, "anything that renders the product not reasonably safe"[64] (i.e. a characteristic of an object which hinders its proper usability).
- *Failure*, as defined in[44] as "a state of inability to perform a normal function". The term is structured and detailed in [57, 30] but relying on an abstract notion of failure without a specific definition.
- *Hazard*, "a potential source of harm"[30].

3 Glossary – A Formalization

We now define a formalizations of the concepts described in Section 2 and depicted in Figure 2. We base our formalization on first order logic (FOL) with the standard truth-value semantics. The choice of this logic is required by the semantics of the concepts formalized afterwards in this section.

We consider a first-order language $\mathcal{L}$ over a signature Σ_P where $P, P', \ldots, P^n$ represent terms, and $\varphi, \varphi', \ldots, \varphi^n$ represent formulas. The syntax is defined as follows.

$$\varphi := P \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \Rightarrow \varphi' \mid \forall P.\ \varphi \mid \exists P.\ \varphi$$

where $\wedge$, $\neg$, $\vee$, and $\Rightarrow$ are connectives representing conjunction, negation, disjunction, and (material) implication respectively; while $\forall$ and $\exists$ represent the standard universal and existential quantifiers (resp.). Finally, the symbol "." is just syntactic sugar. We consider $\sigma \subset \Phi \times \{\top, \bot\}$ as the interpretation function, where Φ is any collection of sentences in $\mathcal{L}$ and $\top$ and $\bot$ represent the concepts of "Tautology/True" and "Contradiction/False" respectively.

Following Figure 2, we start by formalizing the outermost term: Causality.

3.0.1 Causality Principle

We formalize the Causality Principle starting from a K modal logic[33] (i.e. without restrictions on the causality relation between worlds). The standard definition of K modal logic is given in Definition 3.1 in terms of an interpretation function (which we named σ with a slight abuse of notation) defined in Definition 3.2.

Definition 3.1. K Modal Logic – A K-frame is a frame $\mathbf{K} = \langle G, R \rangle$ in the K modal logic where R is the binary relation (i.e. a set of ordered pairs) between possible worlds $R \subseteq G \times G$, where G represents the possible worlds, and $G \neq \emptyset$. An actual world $\omega * \in G$ is assumed. For any proposition P, an interpretation function $\sigma(\omega, P)$ returns the truth value of P; e.g. $\sigma(\omega, P) = \top$ means that P holds in ω. A model is defined as the tuple $M = \langle G, R, \sigma \rangle$.

Definition 3.2. Causality as K Modal Interpretation Function – Causality is (recursively) defined as the modal interpretation function σ, as follows.

($\sigma 0$) if $\sigma(\omega, P) = \top$ then $\omega \models P$
($\sigma 1$) $\omega \models \neg P$ iff $\omega \not\models P$
($\sigma 2$) $\omega \models P \wedge Q$ iff $\omega \models P$ and $\omega \models Q$
($\sigma 3$) $\omega \models \Box P$ iff for any world $\omega' \in G$ if $\omega R \omega'$ then $\omega' \models P$
($\sigma 4$) $\omega \models \Diamond P$ iff there exists a set of worlds $\Omega' \subset G$ such that for any $\omega' \in \Omega'$, if $\omega R \omega'$ then $\omega' \models P$
($\sigma 5$) $\models P$ iff $\omega * \models P$

where truth is defined as necessary with $\Box$ and possible with $\Diamond$.

The causality principle has been defined in its generic form. In fact, the accessibility relation R is free from any axiomatic restriction (e.g. it's not reflexive nor anti-reflexive). We will focus in Section **??** (for our tests) on the application of our theory to CPS system engineering. More detailed case studies (i.e. **tofix:** a CPS as a smart power-grid) will be defined in Section **??**, where the accessibility relation defines in details how the system itself evolves, due to causal relation (i.e. by restricting the causality principle only to those cause-effect that defines the system). Therefore, the definition of R will be specialized in a more strict way based on the application domains and case study.

We note that we have defined the Causality principle without considering it a *threat source* since we lack the concept of intent and maliciousness. Similarly, in the next sections we won't discriminate between Nature and Mankind until we introduce the concepts of maliciousness in Section 4 and then formally define a threat agent in the same section.

3.0.2 Agents: Mankind and Nature

Mankind and Nature are defined in Section 2 as two abstract agents, both as collections (i.e. an abstract type that does not imply a specific implementation) of their sub-agents (i.e. humans for Mankind and plants, animals, &c. for

Table 1: RCC3, RCC5, and RCC8 relations between Regions X, Y and Z

RCC3	**RCC5**	**RCC8**	**Name**	**Notation**	**Definition**
			Connects with	$C(X,Y)$	$X \subseteq Y$
			Disconnected from	$\neg C(X,Y)$	$X \nsubseteq Y$
			Part of	$P(X,Y)$	$\forall Z\ C(Z,X) \rightarrow C(Z,Y)$
			Overlaps	$O(X,Y)$	$\exists Z\ P(Z,X) \wedge P(Z,Y)$
●			**Overlaps Not Equal**	$ONE(X,Y)$	$O(X,Y) \wedge \neg EQ(X,Y)$
●	●	●	**Equal to**	$EQ(X,Y)$	$P(X,Y) \wedge P(Y,X)$
●	●	●	**DiscRete from**	$DR(X,Y)$	$\neg O(X,Y)$
	●	●	**Partial-Overlap**	$PO(X,Y)$	$O(X,Y) \wedge \neg P(X,Y) \wedge \neg P(Y,X)$
	●		**Proper-Part-of**	$PP(X,Y)$	$P(X,Y) \wedge \neg P(Y,X)$
	●		**Proper-Part-of-*i*nverse**	$PPi(X,Y)$	$P(Y,X) \wedge \neg P(X,Y)$
		●	**Externally Connected**	$EC(X,Y)$	$C(X,Y) \wedge \neg O(X,Y)$
		●	**Tangential PP**	$TPP(X,Y)$	$PP(X,Y) \wedge \exists Z\ [EC(Z,X), EC(Z,Y)]$
		●	**Tangential PPi**	$TPPi(X,Y)$	$TPP(Y,X)$
		●	**Non-Tangential PP**	$NTPP(X,Y)$	$PP(X,Y) \wedge \neg\exists Z\ [EC(Z,X), EC(Z,Y)]$
		●	**Non-Tangential PPi**	$NTPPi(X,Y)$	$NTPP(Y,X)$

Nature). Similarly to[71], we define Mankind and Nature, and any other agent in the reminder of this article, as a meronomy (an hierarchy of Part-Whole relations) based on a standard definition of mereology, i.e. based on the definition of Parthood relation between *Parts*. However, shall consider different types of Part; so, we extend the mereology to a mereo-topology[72, 84, 62], to increase the number of different types considered and to generalize the relations between Parts (as in Table 1). For the sake of readability, we use the term *Region* both to refer to a mereological Part and to a topological Region. The choice of mereotopology is also correlated to the objective of defining a formal ontology, which we use to define the (formal) semantics of the terms (Parts) in Section 2, and of the concepts of safety and security (whole). We aim at creating a meronomy instead of the taxonomies such as the one provided in[77, 63] or instead of the poorly justified CVSS[56] scoring system.

A mereotopology, as defined e.g. in[62], is an ordered mathematical structure where the basic relation between Regions is the reflexive and symmetric**FIX**[12] *Parthood* relation $\subseteq$.

Definition 3.3. Parthood – Given any pair of mereotopological Regions X and Y,

1. Reflexivity: $\forall X.\ (X \subseteq X)$
2. Symmetry: $\forall X, Y.\ (X \subseteq Y \Rightarrow Y \subseteq X)$

As later defined in Definition 3.5, the Parthood relation orders a universe of agents $\mathcal{A}$ by defining the so called *Connects with* (see in Table 1) relation between Regions. We want this universe $\mathcal{A}$ to be expressible in FOL. In this way, we can reason both on the constituent of security (i.e. its terms and agents defining a system where security needs to be considered), and on the evolution of those constituent w.r.t. cause-effects relations according to the

[12]**mr:** I guess it must be monotonic as defined in[62] but I don't find it consistently in other papers.

modal structure of causality we defined in Section 3.0.1. This will allow us (in Section **??** and Section **??**) a better**FIX**[13] positioning w.r.t. risk assessment technologies (which most often reason on the constituent of a system design), and protocol verification tools (which requires some formalization of a cause-effect relation, e.g. in Linear Temporal Logic).

As we argued in Section 2, we must correlate the definition of agent (i.e. Mankind and Nature) in Definition **??** to the mathematical structure of the logic that defines them. We express Mankind and Nature as formulas over the theory of mereology and then in terms of mereotopological Regions, extending the interpretation function σ, to include a formal theory of mereotopology. We use the Region Connection Calculus (RCC), as defined in [52, 34], to provide an axiomatization of the spatial concepts and relations in first-order logic to correlate the algebraic structure to mereology. In its broader definition, the RCC theory is composed by eight axioms, and is known as RCC8. In the text, for brevity, we will often focus only on RCC5 (without loss of generality) by not considering tangential connections between spatial Regions. We discuss the choice of RCC5 in more detail in Appendix **??**. In Table 1, we summarize the axioms of the Region Connection Calculus (see, e.g., [35]).

Definition 3.4. RCC axiomatization – For any X, Y pair of Regions in a mereotopology:

(σ6) $\sigma(X \subseteq Y)$ iff $[\sigma(X \subseteq X) = \top$, and $\sigma(X \subseteq Y) = \bot$ or $\sigma(Y \subseteq X) = \top]$**FIX**[14]
(σ7) $\omega \models C(X, Y)$ iff $\omega \models X \subseteq Y$
(σ8) $\omega \models \neg C(X, Y)$ iff $\omega \not\models X \subseteq Y$
(σ9) $\omega \models P(X, Y)$ iff $\omega \models \forall Z.\ C(Z, X) \Rightarrow C(Z, Y)$
(σ10) $\omega \models O(X, Y)$ iff $\omega \models \exists Z.\ P(Z, X) \wedge P(Z, Y)$
(σ11) $\omega \models EQ(X, Y)$ iff $\omega \models P(X, Y) \wedge P(Y, X)$
(σ12) $\omega \models DR(X, Y)$ iff $\omega \models \neg O(X, Y)$
(σ13) $\omega \models PO(X, Y)$ iff $\omega \models O(X, Y) \wedge \neg P(X, Y) \wedge \neg P(Y, X)$
(σ14) $\omega \models PP(X, Y)$ iff $\omega \models P(X, Y) \wedge \neg P(Y, X)$
(σ15) $\omega \models PPi(X, Y)$ iff $\omega \models P(Y, X) \wedge \neg P(X, Y)$

where Z is a mereotopological Region.

Definition 3.5. Agent: Mankind or Nature – An agent $a \in \mathcal{A}$ is a tuple $\langle rcc(\chi', \chi''), \ldots, rcc(\chi^{n-1}, \chi^n)\rangle$ of RCC relations rcc over mereotopological Regions $\chi', \ldots, \chi^n \subseteq$ X.

Currently, we do not distinguish between Mankind and Nature (since we still lack of the definition of "malicious intent", which is defined in Section 4) and we have defined them as two generic agents.

As depicted in Figure 2, Causality, Mankind, and Nature have a dashed border representing their correlation in terms of cause-effect and then in terms

[13] **mr:** Can we prove this by construction?

[14] **mr:** would it be better/clearer or just correct to write $\sigma(X \subseteq Y)$ iff $\sigma(X \subseteq X \wedge [X \subseteq Y \vee Y \subseteq X]) = \top$

of formal structure which defines them: Modal Logic. Vulnerability, Defect, Incident, Accident, and Threat, similarly, are correlated (depicted as a solid border) in terms of underlying formal structure: Mereotopology.

3.0.3 Regions: Vulnerability and Defect (and Weakness)

As defined in Section 2, a Vulnerability or a Defect is a *Weakness*. As an example, a categorization of weaknesses is given in[21] with 808 weaknesses categorized as "Research Concepts", distributed as follows:

- Incorrect Calculation - (682)
- Incorrect Access of Indexable Resource ("Range Error") - (118)
- Use of Insufficiently Random Values - (330)
- Improper Interaction Between Multiple Correctly-Behaving Entities - (435)
- Improper Control of a Resource Through its Lifetime - (664)
- Insufficient Control Flow Management - (691)
- Protection Mechanism Failure - (693)
- Incorrect Comparison - (697)
- Improper Check or Handling of Exceptional Conditions - (703)
- Improper Enforcement of Message or Data Structure - (707)
- Improper Adherence to Coding Standards - (710)

The definition given by the MITRE in[32] of weakness is: "Software weaknesses are errors that can lead to software vulnerabilities. A software vulnerability, such as those enumerated on the Common Vulnerabilities and Exposures (CVE) List, is a mistake in software that can be directly used by a hacker to gain access to a system or network". The definition is circular if we interpret the word "error" and "mistake" with the same semantics: a weakness is an error that leads to a vulnerability and a vulnerability is a mistake which, in turn, is a weakness. The only difference (between weakness and vulnerability) seems to be that one can consider weakness as a ground term and state that a vulnerability is caused by a weakness, i.e. $\Omega, W \models \Diamond V \wedge W$ where W, V are Regions of Weaknesses and Vulnerabilities (resp.); accepting the hierarchy in the CWE[13] as ground truth. Similarly, we consider the CVE[14] (a database of Vulnerability) or the CVE reported in the NVD as a ground truth.

Definition 3.6. Region: Weakness and Vulnerability – A Region $\chi \subseteq a$ of an agent $a \in \mathcal{A}$, is defined as Weakness W (i.e. representing a weakness introduced by the agent into any phase of production, e.g. of the secure process development life-cycle, of any system or subsystem) iff there exists $\omega \in G$ such that $\omega, W \models \Diamond \exists W', V'.\ rcc(W', V') \wedge \neg DR(W', V')$, where $W' \subseteq W$, and $V' \subseteq V$ is a Region of an agent $a' \in \mathcal{A}$.

Example 1. *CWE-116: Improper Encoding or Escaping of Output[43] –*

- *Description: The software prepares a structured message for communication with another component, but encoding or escaping of the data is either missing or done incorrectly. As a result, the intended structure of the message is not preserved.*

- *Example: This code displays an email address that was submitted as part of a form. Example language JSP.*

  ```
  <% String email = request.getParameter("email"); %>
  ...
  Email Address: <%= email %>
  ```

  The value read from the form parameter is reflected back to the client browser without having been encoded prior to output, allowing various XSS attacks (CWE-79).

- *Observed Examples*

  – *CVE-2008-4636[19]: OS command injection in backup software using shell metacharacters in a filename; correct behavior would require that this filename could not be changed.*

  – *CVE-2008-0769[17]: Web application does not set the charset when sending a page to a browser, allowing for XSS exploitation when a browser chooses an unexpected encoding.*

  – *CVE-2008-0005[15]: Program does not set the charset when sending a page to a browser, allowing for XSS exploitation when a browser chooses an unexpected encoding.*

  – *CVE-2008-5573[20]: SQL injection via password parameter; a strong password might contain &*

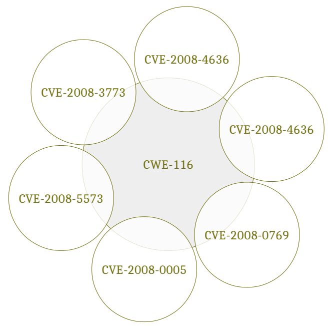
  – *CVE-2008-3773[18]: Cross-site scripting in chat application via a message subject, which normally might contain & and other XSS-related characters.*

  – *CVE-2008-0757[16]: Cross-site scripting in chat application via a message, which normally might be allowed to contain arbitrary content.*

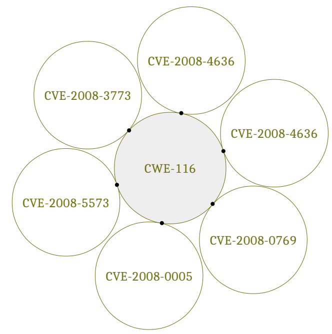
Definition 3.6 states that CWE-116 is a weakness iff there exists a world in K Modal Logic, representing the system in which this weakness exists, such that the natural evolution of this system (i.e. formalized by the causality principle) make it possible to reach another state of the system (i.e. another, accessible world) where there exist a vulnerability that can be implemented from CWE-116 (and this relation is not DR). The CWE website proposes the connection between CWE-116 and, for example, CVE-2008-5573; a vulnerability of the login subsystem of the Poll Pro v2.0[81] system. The formal relation between the two is given as a link (i.e. URI) between the CWE and the CVE, the description of the relation is not defined but it is supposed to be inferred from the descriptions of the CWE-CVE. We can formally represent this link as the ONE connection in RCC3, depicted in Figure 3a, or EC connection in RCC8 (Figure 3b).

FIX[15]

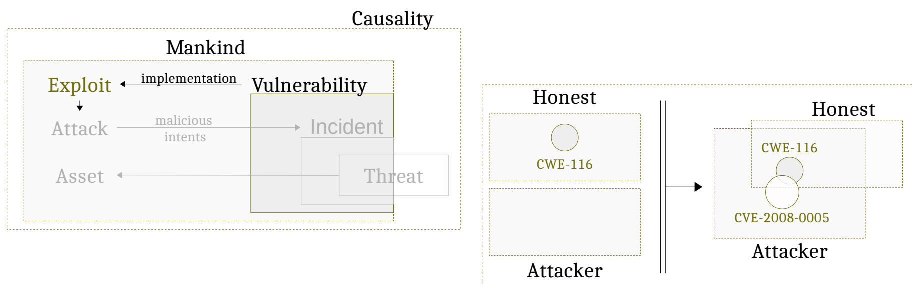
[15] **mr:** shouldn't we consider the agent who introduces the weakness as dishonest? don't we say this in part 1?



(a) Representation with RCC3



(b) Representation with RCC8



(c) Representation in K-Modal Logic

Figure 3: Relations between CWE-116 and correlated CVEs

It is interesting to note that the CWE-CVE relation expresses the correlation between weaknesses and vulnerabilities in the most simple form, such that we can formalize all the relation as ONE in RCC3 or EC in RCC8. To express a more complex relation between the two we shall analyze the definition of CWE-116. This is related to the Weakness-Vulnerability-Incident process (i.e. to the details on the implementation/realization in Figure 2) that we analyze in the next section, Section 3.0.4.

Definition 3.7. Region: Vulnerability and Defect – A region $\chi \subseteq a$ of an agent $a \in \mathcal{A}$ is called *Vulnerability* if the agent a is referred to as Mankind, *Defect* if the agent is referred to as Nature.

3.0.4 Process: Incidents and Accidents

In our informal definition depicted in Figure 2, an Incident is generated through a process caused by a Vulnerability (which, in turn, is caused by a Weakness). Symmetrically, Nature has a process from Defect to Accidents. We start by analyzing the process generated by Mankind, which is divided into the following phases (as shown in Figure 5):

1. An agent (Mankind or Nature) defines a system, e.g. by following an engineering process for creating a new system, or for redesigning or improving

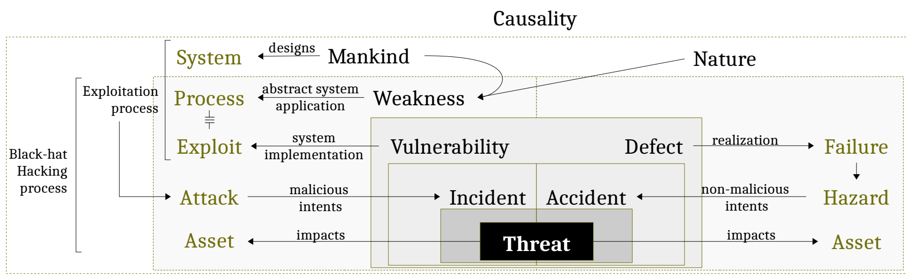


Figure 4: Updated overview security and safety keywords, where Process has to be intended as "Abstract Exploitation Process"

an existing system (e.g. a component or a collection of components)

2. The dogmatic definition of a system contains a Weakness (i.e. the agent, Mankind or Nature, introduces an unintended behavior), for example as an un-intended behavior or as an architectural fallacy. The Weakness can be applied to the abstract definition of the system.
3. The Weakness can be defined as a malicious process for a system, called Vulnerability (or Defect for Nature)
4. The process expressed by the Vulnerability can be made operational for a specific implementation of the system (i.e. implemented by Mankind or realized by Nature)

Those additional details are depicted in Figure 4, an update of Figure 2.

Definition 3.8. Exploitation Process – An *Exploit* is an *implementation* of a Vulnerability, where a Vulnerability describes how to transform a Weakness into an abstract malicious process, which can be applied[16] to one or many systems. So, the act of an agent (Mankind) of "exploiting a Vulnerability" is the process of making the Vulnerability operational, by implementing the Vulnerability w.r.t. a specific target system. Whenever the agent is Nature, the implementation is more broadly considered a realization[17].

The Common Attack Pattern Enumeration and Classification[12] (CAPEC) online service provides a "community resource" which aims at "identifying and understanding attacks". The MITRE writes on the CAPEC homepage that "Understanding how the adversary operates is essential to effective cyber security. CAPEC helps by providing a comprehensive dictionary of known patterns of attack employed by adversaries to exploit known weaknesses in cyber-enabled capabilities. It can be used by analysts, developers, testers, and educators to advance community understanding and enhance defenses.". We can extrapolate two major concepts:

[16] Meaning that the malicious process, i.e. any structured procedure such as a protocol, can be somehow made operational, e.g. implemented

[17] "The state of being realized"[45]

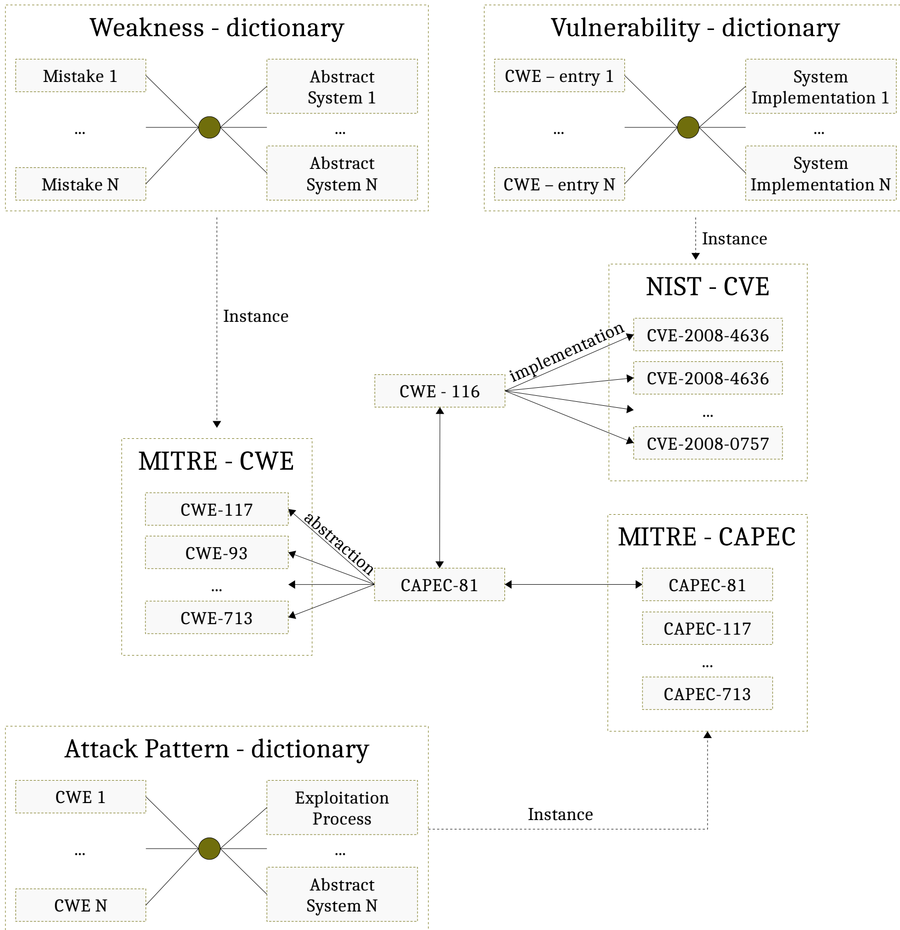


Figure 5: Cybersecurity online dictionaries and connections

- While understanding the exploitation process is the aim of the CWE and CVE initiatives, understanding the hacking process is the aim of the CAPEC initiative.
- CAPEC provides a *dictionary* of *known* attack patterns. As stated in details in Section 2, this dictionary contains empirical evidences and should not be used to induce a cybersecurity theory; but it can be used to validate a theory.

Example 2. *The CWE-116 doesn't just relate the Weakness to a number of Vulnerability in the CVE but also relates the Weakness to a number (4 for the CWE-116) of abstract attack path in the CAPEC:*

- *CAPEC-104[8] Cross Zone Scripting*

- *CAPEC-73[9] User-Controlled Filename*
- *CAPEC-81[10] Web Logs Tampering*
- *CAPEC-85[11] AJAX Fingerprinting*

Each CAPEC entry has a reference to one or many CWE, including CWE-116.

After the exploitation process, as depicted in Figure 5, another process starts and transforms a Vulnerability in an Incident by Attacking the system with a malicious Hack, as follows.

5. An agent (Mankind[18]) the exploitation process can be used to hack a system. We use the term hack (as hacking) to stress that nothing prevents this process from being honest, however, for the sake of simplicity, we focus on Attacks which are malicious by definition.
6. An Attack is applied to a system with malicious intent by an agent
7. The application of the attack results in an Incident which impacts, i.e. poses a Threat to, an Asset

Definition 3.9. Black-hat Hacking Process An *Attack* is the act of making operational an *Exploitation Process* causing an Incident.

Example 3. *CAPEC-81[10] Web Logs Tampering –*

- *Description: An attacker is able to cause a victim to load content into their web-browser that bypasses security zone controls and gain access to increased privileges to execute scripting code or other web objects such as unsigned ActiveX controls or applets. This is a privilege elevation attack targeted at zone-based web-browser security. In a zone-based model, pages belong to one of a set of zones corresponding to the level of privilege assigned to that page. Pages in an untrusted zone would have a lesser level of access to the system and/or be restricted in the types of executable content it was allowed to invoke. In a cross-zone scripting attack, a page that should be assigned to a less privileged zone is granted the privileges of a more trusted zone. This can be accomplished by exploiting bugs in the browser, exploiting incorrect configuration in the zone controls, through a cross-site scripting attack that causes the attackers' content to be treated as coming from a more trusted page, or by leveraging some piece of system functionality that is accessible from both the trusted and less trusted zone. This attack differs from "Restful Privilege Escalation" in that the latter correlates to the inadequate securing of RESTful access methods (such as HTTP DELETE) on the server, while cross-zone scripting attacks the concept of security zones as implemented by a browser.*

[18]For the sake of readability we focus only on Mankind as an agent

- *Execution Flow:*

  1. *Explore: Find systems susceptible to the attack: Find systems that contain functionality that is accessed from both the internet zone and the local zone. There needs to be a way to supply input to that functionality from the internet zone and that original input needs to be used later on a page from a local zone.*
  2. *Experiment: Find the insertion point for the payload: The attacker first needs to find some system functionality or possibly another weakness in the system (e.g. susceptibility to cross site scripting) that would provide the attacker with a mechanism to deliver the payload (i.e. the code to be executed) to the user. The location from which this code is executed in the user's browser needs to be within the local machine zone.*
  3. *Exploit: Craft and inject the payload: Develop the payload to be executed in the higher privileged zone in the user's browser. Inject the payload and attempt to lure the victim (if possible) into executing the functionality which unleashes the payload.*

- *Prerequisite: The target must be using a zone-aware browser.*

- *Consequences:*

  - *Integrity: Modify Data*
  - *Confidentiality: Read Data*
  - *Confidentiality, Access Control, Authorization: Gain Privileges*
  - *Confidentiality, Integrity, Availability: Execute Unauthorized Commands*

The formalization of the Exploitation and Black-hat Hacking processes is the formalization of two foundational cybersecurity concepts which lay the basis for the definition (and formalization) of the cybersecurity theory in the next Section (Section 4). The formal definition of those processes shall:

- in Section 4, formally define a system as a structure of agents, and
- in Section **??**, correlate the formal definition of agents (i.e. the Definition 3.5) with the Causality principle (i.e. the formalization of the Kripke structure in Definition 3.1).

4 $\mathcal{ABF}$ – A Security Theory

In order to address the problem raised by Herley, we shall define how to distinguish between a secure and an insecure system. While most of the literature (and this paper so far) have correlated the problem of in-security to the maliciousness of agents interacting with the system, we show that security doesn't seem to stem from a malicious nature but, rather, insecurity raises from the lack of well-defined security requirements for the design process of a system. We argue that the high number of security vulnerability reported today are simply the realization of potential system configurations, which deviate from the nominal behavior just because the *intended behavior* **FIX**[19] of the system is not precisely defined in the specification at the very early stages of the engineering process. A reference engineering process (so called, system security lifecycle of a product) is described, in details, in Section **??**. However, for the sake of simplicity, the reader can assume for this section that we are currently dealing with just the relation between the specification of a system and its design.

Given that we apply our theory to *systems engineering* and that the two foundational processes (exploitation and hacking) are based on the concept of a system, we shall begin by defining what a system is, in its abstract form.

4.1 A Reference Model for Cybersecurity Engineering

In order to describe the $\mathcal{A}, \mathcal{B}, \mathcal{F}$-theory we briefly introduce the engineering concepts (detailed and extended in Section 5) underlying the $\mathcal{A}, \mathcal{B}, \mathcal{F}$-theory; namely, the first three stages of the, so called, engineering V-Model. Our definition is partial and we only use it for the sake of simplicity and to better present the theory.

The development process of a new product (a CPS in our case) should follow an engineering model. Our reference engineering process (inspired by the waterfall V-Model[**waterfallvmodel**]), as depicted in Figure 6, is structured as the following ordered list of abstraction refinement steps.

1. The process starts with the definition of a *specification* where the general requirements of the CPS are defined (e.g. functional, physical, network, and design). As an example, an RFC of a communication or security protocol can be considered as the result of this first phase (e.g. see RFC-1[**rfc1**]. In this first phase, we consider the architectural aspects of a CPS such as the draft physical and functional architectures.

2. The process then continues with the definition of a design where the requirements of the previous phase holds; detailing the logic of the physical (e.g. the physical-layer encoding of the information) and functional architectures (e.g. the protocol logic of communication over a number of potential sessions).

3. Finally, the design is implemented as hardware (HW) for the physical architecture and software (SW) for the functional architecture.

[19] **mr:** maybe "nominal" instead of "intended" is better and behavior can be confused with the relation between assertions and beliefs which is not what we mean

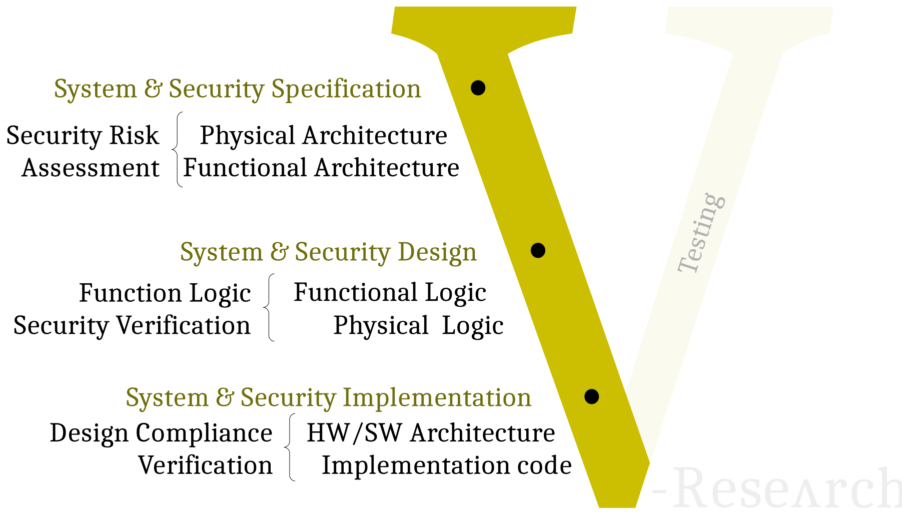


Figure 6: Cybersecurity Engineering Life-cycle

4.2 Abstract System

The ISO/IEC/IEEE 15288:2015 (System Life Cycle Processes) provides a definition of system as “A combination of interacting elements organized to achieve one or more stated purposes.”[75]. Therefore, a system can be considered as a single agent where its interacting elements are the constituents of the agent itself. For the sake of simplicity we first define a system as an agent and then extend the definition to a “combination of interacting” agents. We use the same process of [71] but over slightly different terms. The differences and commonalities with [71] are described in Appendix **??**.

There is no agreement between the research communities (e.g. Multi-Agent-System, Epistemic Logic) on which are the constituent of an agent as a system. However, the same ideas revolved around for thousands of years. Some relevant examples for our objective are the following.

- In [41], Hintikka describes the difference between Knowledge and Belief (as epistemological concepts), and the whole Doxastic logic defines in details how Beliefs can be formalized.
- In [42], Hintikka describes the concept of Information and the difference with Knowledge and Belief.
- In [29], the author states: “The logical criterion also may be used in three senses – of the agent, or the instrument, or the “according to what”; the agent, for instance, may be a man, the instrument either sense perception or intelligence, and the “according to what” the application of the impression “according to” which the man proceeds to judge by means of one of the aforesaid instruments.”

- In [71], the authors defines an agent as a tuple of Assertions, Beliefs, and Facts.

If we consider [29], the author applies a "logical criterion" to an agent, for instance a man, for instance a device in a system (or a system) as in [71]. The instrument identified is sense perception, because a man is considered as example, but we can abstract from the concept of instrument and consider any received information, where information is considered as in [42]; which can be summarized by the following axiom: "Information is specified by specifying which alternatives concerning the reality it admits and which alternatives excludes". This means, for our argument, that considering a propositional variable, which admits the two alternatives True/False, its information is defined as Believed to be True/False and not Believed to be the opposite. Finally, in [29] the author states that the logical criterium can be judged according to a frame of reference. For our argument, as in [71], we define this concept as Knowledge (or Facts); where knowledge is defined, as in [79], as a set of proposition known by an agent, such that: (i) knowledge requires belief, (ii) knowledge require truth, (iii) knowledge must be *properly justified*.

Before detailing what "properly justified" means, we discuss the mentioned relation between agent and belief and we introduce a more formal definition of knowledge and belief. For our argument, the only objective of Information is to exchange beliefs between agents. Due to the definition of Information and then with its relation to the probabilistic correlation to truth/reality, we consider information in relation with an agent's beliefs. Similarly, we consider beliefs to define the actual behavior of an agent or a system. On the other hand, Knowledge drives the nominal behavior of an agent or a system. This view is not considered in most (if not all) the approaches to protocol verification and the Dolev-Yao theory is usually applied to a representation/abstraction of a protocol that only considers knowledge and transfer of knowledge (i.e. if an agent sends a message, the recipient knows that message). However, the security of a system is tightly related to the difference between the nominal behavior and the actual (e.g. the implemented) one.

Definition 4.1. System State and State-Space – The state of a system (or a sub-system) s is identified with an instance of an agent state (see Definition 3.5) such that $s = \langle rcc(\mathbb{K}, \mathbb{B}), rcc(\mathbb{K}, \mathbb{I}), rcc(\mathbb{B}, \mathbb{I})\rangle$, where $\mathbb{K}, \mathbb{B}, \mathbb{I}$ are Regions of Knowledge, Beliefs, and Information respectively, and rcc is a specific relation between the Regions it predicates on. The state-space of a system is then defined by the different rcc relations between the three pairs of Regions defining the system and all the sub-systems.

Therefore, a system state space can be seen as a superposition of multiple agent states where knowledge, beliefs, and information are related between each other (in a mereotopological space) through all the admissible RCC relations. **FIX**[20] A *system state* (as an instance of the system state space) is a specific con-

[20] **mr:** are the next lines redundant?

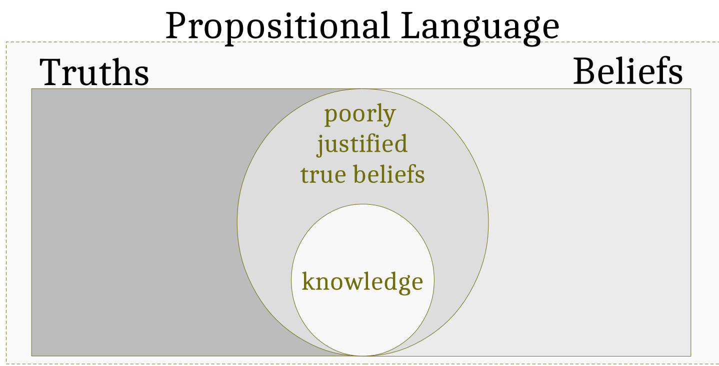


Figure 7: Informal representation of Knowledge and Belief

figuration of the *rcc* relations between the three regions. Therefore, a collection of system states defines a collection of choices for those *rcc* relations.

The difference between Knowledge and Belief is depicted in Figure 7 (see [47]). However, according to[82], Knowledge[21] as an epistemological concept is difficult to formally define. Similarly for the concept of information, Hintikka in [42] states that "A purely logical definition of information is impossible". In this work, however, we are not interested in how knowledge, information, or belief can be precisely formalized from an epistemic standpoint. We assume that a semantic of a correct (i.e. commonly believed to be true) definition of epistemic knowledge exists, for example the one given in[41] by Hintikka, and we then define knowledge in terms of the Kripke structure defined in Definition 3.1; similarly for Belief.

Definition 4.2. Knowledge – Given an abstract collection of Agents Ag, and the modal operator K_a (where $a \in Ag$), Knowledge is defined as a region of predicates known by an agent $\mathbb{K}_a = \bigcup_\Phi K_a\varphi$ (where Φ is the collection of all the propositions known by a). Given a proposition P, we extend the semantics of the Causality structure with:

(σ16) $\omega \models K_a P$ iff $\omega' \models P$ for all ω' such that $\omega R \omega'$.

Definition 4.3. Belief – Given an abstract collection of Agents Ag, and the modal operator B_a (where $a \in Ag$), Belief is defined as a region of predicates believed by an agent $\mathbb{B}_a = \bigcup_\Phi B_a\varphi$ (where Φ is the collection of all the propo-

[21] "*Theaetetus*: [...] He said that knowledge was true opinion accompanied by reason, but that unreasoning true opinion was outside of the sphere of knowledge; and matters of which there is not a rational explanation are unknowable – yes, that is what he called them – and those of which there is are knowable. [...] *Socrates*: [...] the primary elements of which we and all else are composed admit of no rational explanation; for each alone by itself can only be named, and no qualification can be added, neither that it is nor that it is not, for that would at once be adding to it existence or non-existence, whereas we must add nothing to it, if we are to speak of that itself alone. [...]" Plato – Theaetetus 201 [60]

sitions believed by a). Given a proposition P, we extend the semantics of the Causality structure with:

(σ17) $\omega \models B_a P$ iff $\omega \models \neg K_a \neg P$ (i.e. the agent a considers P possible) and $\omega' \models P$ for all ω' such that $\omega R \omega'$.

Definition 4.4. Information – Given an abstract collection of Agents Ag, and the modal operator I_a (where $a \in Ag$), Information is defined as a region of beliefs asserted by an agents a. $\mathbb{I}_a = \bigcup_\Phi I_a \varphi$ (where Φ is the region of all the propositions believed and asserted by a). Given a proposition P, we extend the semantics of the Causality structure with:

(σ18) $\omega \models I_a P$ iff $\omega \models \mathbb{B}_a P$ (i.e. the agent a considers P possible), $\omega \models \neg \mathbb{B}_a \neg P$

4.3 Operational System

When dealing with a specification of a system (i.e. the design if the system for a specific operation (as a generic operating/operational system), considering abstract and general concepts such as knowledge, belief, or information may result to be too abstract for the objectives of the overall engineering process (e.g. verification and validation). As an example, we are usually not interested in information in abstract but to the transfer of that information, which we call Assertions, made by a subset of agents. Therefore, we now define how we consider an agent for the engineering of CPS.

1. We consider Information only when the intention of exchanging that Information is from a sender to recipient is defined; and we call it *Assertion*
2. Similarly, the portion of Beliefs we consider for system engineering is the one that builds (input) or describes (output) the behavior of an agent (strategy rules [22]), and
3. We consider a set of axiomatic *Facts* (definitory rules**FIX**[23]) instead of considering the more general epistemic definition of Knowledge. Specifically, Facts describes:
   - The functional architecture of each agents
   - The physical/structural (HW/SW) architecture of each subsystem of agents and agent within a subsystem

[22] "The logical structure of information is one of the most basic and one of the most basic and one of the simplest thing in the wide and wonderful world of logical analysis. This point can be put in a deeper perspective. A distinction [...] ought to be made [–] between two kinds of rules (or principles) in any strategic activity like knowledge seeking. On the one hand you have the rules that define the game, e.g. how chessmen are moved on a board. The can be called *definitory* rules. They must be distinguished from rules [...] that deal with what is better and what is worse in the game in question. Definitory rules do not say anything about this subject. Rules which do can be called *strategic rules*" – Hintikka in [42]

[23] **mr:** facts are definitory rule, as they don't define how a real system is finally implemented (curently) but how it should, trugh a series of requirements which may not be respected, through insecurity, in the implemented system

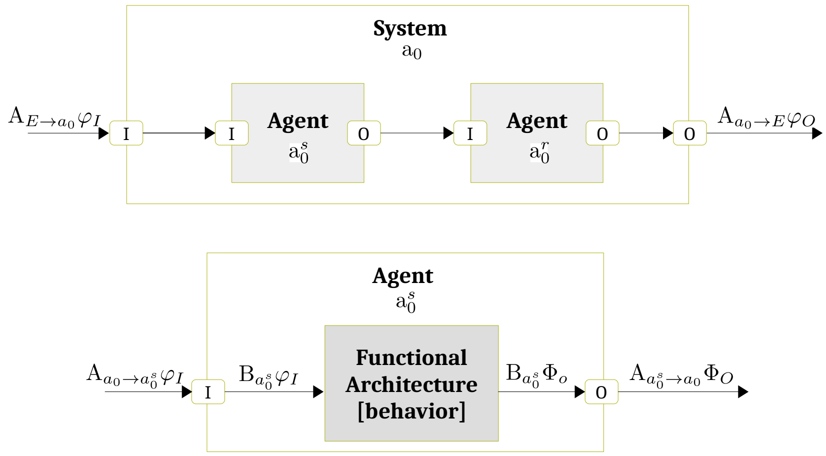


Figure 8: Example of system and agent structure

- Other important aspects of the system that we describe afterwards in this paper, such as the assets and security properties

We give a graphical representation of (sub-)system and agent in Figure 8.

We note here (and describe with more details afterwards in this paper) that the data flow can be defined as a transfer of Beliefs through Assertions (i.e. the Beliefs flow).

Definition 4.5. Assertion – An assertion (**tofix:** subtype of Information) of is an intended transfer of beliefs between two agents a and b such that

(σ19) $\omega \models \mathcal{A}_{a\to b}P$ iff $\omega \models \mathbb{B}_a P$, $\omega \models \neg\mathbb{B}_a\neg P$, $\omega' \models rcc(\mathbb{B}_a P, \mathbb{B}_b P)$, for all ω' such that $\omega R \omega'$.

Definition 4.6. Behavior – The behavior (**tofix:** subtype of Beliefs) of an agent is defined as the transformation process (e.g. defined as a protocol, or a functional architecture) that determines the output-beliefs based on input-beliefs and vice versa (where input-beliefs and output-beliefs are beliefs taken as input or output respectively).

(σ20) $\omega \models \bigcup_\Phi B_s\varphi$ and $\omega \models \mathcal{B}_a[\bigcup_{s\in S} B_s P_s]$ iff $\omega \models \mathbb{B}_a P'$,
where a is an agent, S is a set of agents such that $a\neg \in S$, and the calculation of P' is deteministic.

Definition 4.7. Fact – Facts are a **tofix:** sub-region/subtype of Knowledge $\mathcal{F} = \bigcup_\Phi K_a\varphi$ that predicate in a factive way, e.g. if a knows that P then P, and the region of facts is monotone (no revision).

(σ21) $\omega \models \mathcal{F}_a P$ iff $\omega \models \Box P$.

Definition 4.8. Operational System State – An operational system (or a sub-system) is represented as an agent (see Definition 3.5) and then defined by the resulting state space, such that $s = \langle rcc(\mathcal{F}, \mathcal{B}), rcc(\mathcal{F}, \mathcal{A}), rcc(\mathcal{B}, \mathcal{A})\rangle$, where $\mathcal{A}, \mathcal{B}, \mathcal{F}$, are regions of assertions, behavior (i.e. the beliefs generated by the behavior), and facts respectively.

As already presented in [71] it follows that, defining a system (or an operational system) with a fixed number of regions, there exist an upper-bound to the number of possible configuration of a system, defined by the possible relations between the different regions. For completeness, we report in the next paragraph the calculation done in [71].

Number of different configurations of a system. The general formula to calculate the number of different types of agents is $r^{\binom{n}{k}}$, where r is the number of relations with arity k, between n different sets, where r^e is the number of permutation of r relations over e elements with repetitions, with e being the number of k-ary combinations of n sets, $\binom{n}{k}$. In our case, $\binom{n}{k} = 3$ since we consider 3 sets $(\mathcal{A}, \mathcal{B}, \mathcal{F})$, and all the relations considered in the RCC are binary. Hence, using RCC5 (with five different spatial relations) over three sets, we can theoretically define up to 125 different type of agents. However, only 54 of the 125 (as showed in [34]) combinations are topologically correct with respect to the definition of the relations of RCC5. Generalizing to all the RCCs:

- *RCC3* — theoretical: $3^3 = 27$, correct: 15
- *RCC5* — theoretical: $5^3 = 125$, correct: 54
- *RCC8* — theoretical: $8^3 = 512$, correct: 193

Hence, even if considering a different number of sets than the three $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{F}$ exponentially affects the number of theoretical agents, the application of RCC downscales that number of a factor that ranges from 1.8 to 2.5. In addition, using RCC5 we consider 3.6 times more (different) types of agents than RCC3, but using RCC8 would allow us to consider 3.5 times more different agents.

In the quantitative evaluation of a single agent, depicted in Figure 9, we argue that only 1 configuration represents the nominal (expected) behavior of the agent while the other configurations are either impossible to implement or diverge from the intended nominal behavior. We note that, the numbers reported here do not consider the details of the engineering process and should be considered a limit of an abstract representation of the system. In Section 5 we detail this numeric evaluation to faitfully represent the number of insecurity configurations.

As described beforehand in this section, the collection of Facts shall define the functional and the physical architecture of an agent, but Assets and security properties are defined as Facts, i.e. true. Given that understanding why an Asset is considered as such by the user doesn't necessarily allow us to better qualify an Asset, we consider the fact of "being an asset" as a predicate **tofix:** (or a true function) of a component of a system.

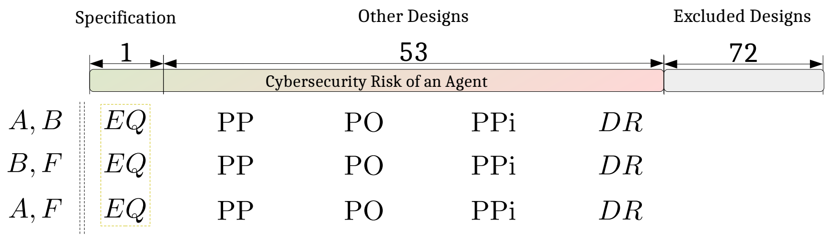


Figure 9: Quantitative representation of the cybersecurity risk for a single agent

Definition 4.9. Asset – For any agent $a \in Ag$ in a collection of Agents Ag, an agent is considered an asset iff

(σ22) $\omega \models \text{asset}(a)$ iff $\omega \models \Box\text{asset}(a)$.

The definition of the security properties is given in Section 5.1

4.4 Qualitative Evaluation of Agent Space in $\mathcal{A}, \mathcal{B}, \mathcal{F}$

While a quantitative analysis reveals how many possible configurations of an agent (i.e. a system) exist w.r.t. the $\mathcal{A}, \mathcal{B}, \mathcal{F}$ theory (i.e. 54/125 in RCC5), a qualitative analysis of the different configurations describes the configurations allowed by the $\mathcal{A}, \mathcal{B}, \mathcal{F}$ theory, and how those configurations can be categorized. In Table 2 we provide the generic composition table of RCC5 over 3 regions instantiated over $\mathcal{A}, \mathcal{B}, \mathcal{F}$, which shows the whole state space for a single agent. The color coding of the table represents the risk function as a risk matrix as we detail afterwards in this section.

A detailed qualitative categorization of agents over a theory similar to the $\mathcal{A}, \mathcal{B}, \mathcal{F}$ one, is presented in this article and described in [71]. The differences in the formalization and of the application requires another in-depth analysis. As depicted in Figure 10, the relation between Facts, and Assertions and Behavior defines the soundness of the design (admissible configurations of Assertions, Behaviors and, in turn, Beliefs) w.r.t. the specification (what the specification mandates, such as by the nature of the physical/functional architectural models). We categorize agents by first analyzing the relations between each pair of Regions defining an agent (i.e. $\mathcal{A}, \mathcal{B}, \mathcal{F}$), and then we categorize the different agents as tuple of the three Regions. For the sake of simplicity, soundness is opposed to non-soundness in the following, however, with the RCC we consider different "degrees" of soundness. For example, in RCC5, if we consider EQ between two Regions as representing soundness, DR over the same Regions represents non-soundness while PP, PO, PPi represents the different degrees of soundness.

rcc($\mathcal{A}$,$\mathcal{B}$) – Collaboration. In order to reason on the relation between assertions and behavior we first need to consider that, by definition, assertions are defined as transfer of information between two agents, e.g., a and b. Therefore,

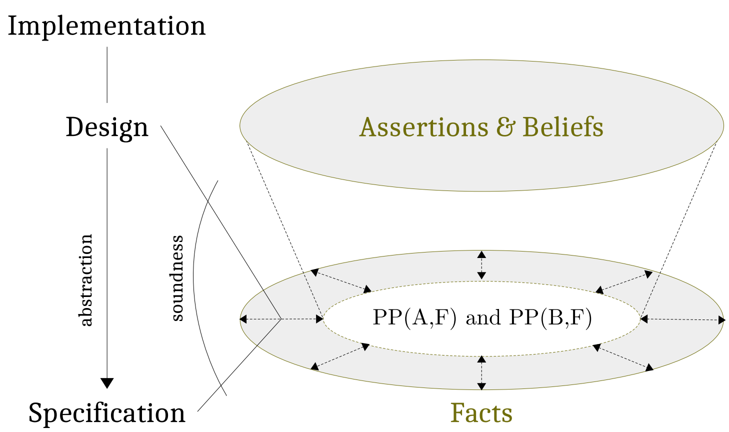


Figure 10: Example relation between Facts, and Assertions and Beliefs

as depicted in Figure 8, an agent has two main categories of assertions, input and output assertions. Given an agent a and a collection of asserted predicates Φ, the Input assertions are those received by a from an agent s acting as a sender, $\mathcal{A}_{s\to a}\Phi$; similarly, output assertions are sent from a to a receiver r, $\mathcal{A}_{a\to r}\Phi$. We shall consider two pairs of regions[24]:

- rcc($\mathcal{A}_{s\to a}$,$\mathcal{B}$), where the relation between Input-assertions and behavior describes the soundness of the execution of the functional architecture w.r.t. input elicitation. With more details, the ideal specification of the functional architecture, along with the expected inputs, defines the functional behavior of an ideal system. If all the inputs (Assertions) are correctly handled in the functional specification (behavior) the specification is sound.

- rcc($\mathcal{A}_{a\to r}$,$\mathcal{B}$), where the relation between behavior and outputs describes the completeness of the behavior defined in the specification w.r.t. the input elicitation. With more details, if all the outputs (assertions) of the functional architecture can be produced, the functional architecture is complete.

rcc($\mathcal{A}$,$\mathcal{F}$) and rcc($\mathcal{B}$,$\mathcal{F}$) – Honesty and Competence. While Assertions and Beliefs generated by the Behavior determines how a system should work, and the relation between the two defines the quality (i.e. the completeness w.r.t the specification) of the agent, the relation of Assertions and Behavior with Facts determines the quality with respect to the nominal (specified) system.

[24] "I am not asserting, as Lotze did, that a relation between X and Y consists of a quality in X and a quality in Y – a view which I regard as quite indefensible. – I assert that a relation Z between X and Y involves the existence in X of the quality "having the relation Z to Y" so that a difference of relations always involves a difference in quality, and a change of relations always involves a change of quality." – Ellis J. McTaggart, The Unreality of Time[55]

Given that Facts defines what is true in the system (e.g. the level of water in a tank may be considered as a Fact, while the reading of a sensor as a Belief), the relation of Assertions an Facts determines the degree of quality between the information circulating in a system (or within an agent) and reality**FIX**[25]. Since the transfer of information through Assertions generates Beliefs, a dishonest agent may circulate false information, generating false Beliefs. We note that it is often implied that the intention behind circulating false information discerns a dishonest and an incompetent agent, however, we consider honesty related to truth rather than related to what is true[26]. This implies that the relation between Beliefs and Facts determines the competence (on the subjected defined by the Facts) of an agent (i.e. the more competent an agent is, the more likely a Belief of that agent is true).

4.4.1 $\mathcal{A}, \mathcal{B}, \mathcal{F}$ Security Enumeration (SE)

The following security requirement for a CPS specification can be summarized:

SE-1 Proper interaction between multiple correctly-behaving agents (in contrast with the "Improper Interaction Between Multiple Correctly-Behaving Entities" defined by the CWE–435 as one of the top "view" of the "research concepts" in [21]) is defined as $EQ(\mathcal{A}_a, \mathcal{B}_a)$ for an agent a

SE-1.1 The equality relation $EQ(\mathcal{A}_{s\to a}, \mathcal{B}_a)$ describes the intended secure behavior as: the beliefs generated by the behavior of the functional architecture shall be complete w.r.t. the specified inputs of the agent. Therefore, *the assertions received by an agent or a system shall be compliant with the expected inputs of the functional architecture.* For example, the inputs of the user of a SW must be sanitized to exclude deviations w.r.t. the expected inputs of the functions implemented in the SW. Another example is the type checking between allowed inputs and expected inputs.

SE-1.2 Similarly, the equality relation $EQ(\mathcal{A}_{a\to r}, \mathcal{B}_a)$ defines that the outputs of an agent a shall be the outputs of the functional architecture.

SE-2 Sufficient Control Flow Management (in contrast with the "Insufficient Control Flow Management" defined by MITRE in the CWE–691 as one of the top "view" of the "research concepts" in[21]) is defined as $EQ(\mathcal{A}, \mathcal{F})$.

SE-3 Correct Calculation (in contrast with the "Improper Calculation" defined by MITRE in the CWE–682 as one of the top "view" of the "research concepts" in[21]) is defined as $EQ(\mathbb{B}, \mathcal{F})$.

[25] **mr:** assertions are reality while facts are dogmatic assertions at specification level, i.e. requirements

[26] "[...] truth exists only in the good man, but the true in the bad man as well; for it is possible for the bad man to utter something true." Sextus Empiricus, Outlines of Pyrrhonism, II-83[29]

We are now in the position to define what a secure system (with respect to the $\mathcal{A}, \mathcal{B}, \mathcal{F}$-theory) is and, based on that definition, what the security risk is and how to quantify it in a risk matrix.

Definition 4.10. Security of a System or an Agent – A secure system is a system where SE-1, SE-2, and SE-3 holds for each agent composing the system.

The ISO 31000 consider risk as the "effect of uncertainty on objectives" and the refers both of positive and negative consequences of uncertainty. Accordingly, we consider risk as follows.

Definition 4.11. Risk – The whole space of potential designs of a specification with respect to the $\mathcal{A}, \mathcal{B}, \mathcal{F}$-theory.

The definition of Risk leads to the risk matrix in Figure Figure 9, defined as follows.

Definition 4.12. Risk Matrix – The risk matrix is a function between the three relations $s = \langle rcc(\mathcal{F}, \mathcal{B}), rcc(\mathcal{F}, \mathcal{A}), rcc(\mathcal{B}, \mathcal{A})\rangle$, where the maximum risk is defined by the DR relation between the three groups of Regions, and the minimum risk by the EQ relation over the same Regions. In between the two extremes, the granularity of possible intermediate configuration is defined by the calculus used (RCC5 in our case).

We note that often the risk matrix is defined by the relation between likelihood and impact. We argue that the likelihood is related to the number of possible non secure (i.e. non maximum secure) configurations of a system and, similarly, the impact.

5 Cybersecurity Engineering Process

> The difference between ideas and reality is like the difference between philosophy and engineering. The work to transform one into the other is scientific research
>
> *V-Research*

Several standards mandates a secure-by-design approach in which cybersecurity shall be considered at the very early stages of the design process. For example,

1. DO-326A – "Airworthiness Security Process Specification" requires a cybersecurity risk assessment of the design and is the "are the only Acceptable Means of Compliance (AMC) by FAA & EASA for aviation cybersecurity airworthiness certification, as of 2019" as pointed out by SAE in [69].

	DR($\mathcal{A}$,$\mathcal{B}$)	PO($\mathcal{A}$,$\mathcal{B}$)	PP($\mathcal{A}$,$\mathcal{B}$)	PPi($\mathcal{A}$,$\mathcal{B}$)	EQ($\mathcal{A}$,$\mathcal{B}$)
DR($\mathcal{B}$,$\mathcal{F}$)	T($\mathcal{A}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$) PO($\mathcal{A}$,$\mathcal{F}$) PP($\mathcal{A}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$) PO($\mathcal{A}$,$\mathcal{F}$) PP($\mathcal{A}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$)
PO($\mathcal{B}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$) PO($\mathcal{A}$,$\mathcal{F}$) PP($\mathcal{A}$,$\mathcal{F}$)	T($\mathcal{A}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$) PO($\mathcal{A}$,$\mathcal{F}$) PP($\mathcal{A}$,$\mathcal{F}$)	PO($\mathcal{A}$,$\mathcal{F}$) PPi($\mathcal{A}$,$\mathcal{F}$)	PO($\mathcal{A}$,$\mathcal{F}$)
PP($\mathcal{B}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$) PO($\mathcal{A}$,$\mathcal{F}$) PP($\mathcal{A}$,$\mathcal{F}$)	PO($\mathcal{A}$,$\mathcal{F}$) PP($\mathcal{A}$,$\mathcal{F}$)	PP($\mathcal{A}$,$\mathcal{F}$)	PO($\mathcal{A}$,$\mathcal{F}$) EQ($\mathcal{A}$,$\mathcal{F}$) PP($\mathcal{A}$,$\mathcal{F}$) PPi($\mathcal{A}$,$\mathcal{F}$)	PP($\mathcal{A}$,$\mathcal{F}$)
PPi($\mathcal{B}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$) PO($\mathcal{A}$,$\mathcal{F}$) PPi($\mathcal{A}$,$\mathcal{F}$)	T($\mathcal{A}$,$\mathcal{F}$)	PPi($\mathcal{A}$,$\mathcal{F}$)	PPi($\mathcal{A}$,$\mathcal{F}$)
EQ($\mathcal{A}$,$\mathcal{F}$)	DR($\mathcal{A}$,$\mathcal{F}$)	PO($\mathcal{A}$,$\mathcal{F}$)	PP($\mathcal{A}$,$\mathcal{F}$)	PPi($\mathcal{A}$,$\mathcal{F}$)	EQ($\mathcal{A}$,$\mathcal{F}$)

Table 2: RCC5 composition table over 3 regions. The results show that there exist 54 possible relations and the coloring anticipates the ideal risk matrix (green the secure state with low risk, red the high risk state, and a gradient of medium risk states). T($\mathcal{A}$,$\mathcal{F}$) = {DR($\mathcal{A}$,$\mathcal{F}$), PO($\mathcal{A}$,$\mathcal{F}$), PP($\mathcal{A}$,$\mathcal{F}$), PPi($\mathcal{A}$,$\mathcal{F}$), EQ($\mathcal{A}$,$\mathcal{F}$)}

2. NIST 800-82 [80] – “guide to Industrial Control System (ICS) Security”
3. J3061:2016-1 [68] – “Cybersecurity Guidebook for Cyber-Physical Vehicle Systems” defines “set of high-level guiding principles for Cybersecurity as it relates to cyber-physical vehicle systems” and states that “incorporate Cybersecurity into cyber-physical vehicle systems from concept phase through production, operation, service, and decommissioning”

Companies who develop CPS (e.g. aerospace systems or elevator systems), must adhere to strict regulations and quality assurances processes. Introducing tools and procedures in their processes requires a detailed and justified overview on how those tools and procedures can be used. In Figure 11, we provide an overview of a process that consider cybersecurity in the specification, design, and implementation stages. The overall process start with the specification of the architectural (both physical and functional) requirements from which the Weaknesses are automatically identified. Based on the Assets, architectures, and the Weaknesses the cybersecurity risk is calculated based on the $\mathcal{ABF}$-theory. An iterative process between the user (e.g. the engineer) and the V-SecRAmodule of the ABFtoolallows the user to move to the design phase when the risk level is considered acceptable. The design phase starts by automatically mapping the Weaknesses into Vulnerabilities which are, in turn, fed into the V-DesignVerifmodule along with the final specification. The Module returns a list of Vulnerabilities and potential Mitigations. The user then procees in implementing the CPS and the SW/HW choices can be tested by the V-ATG.

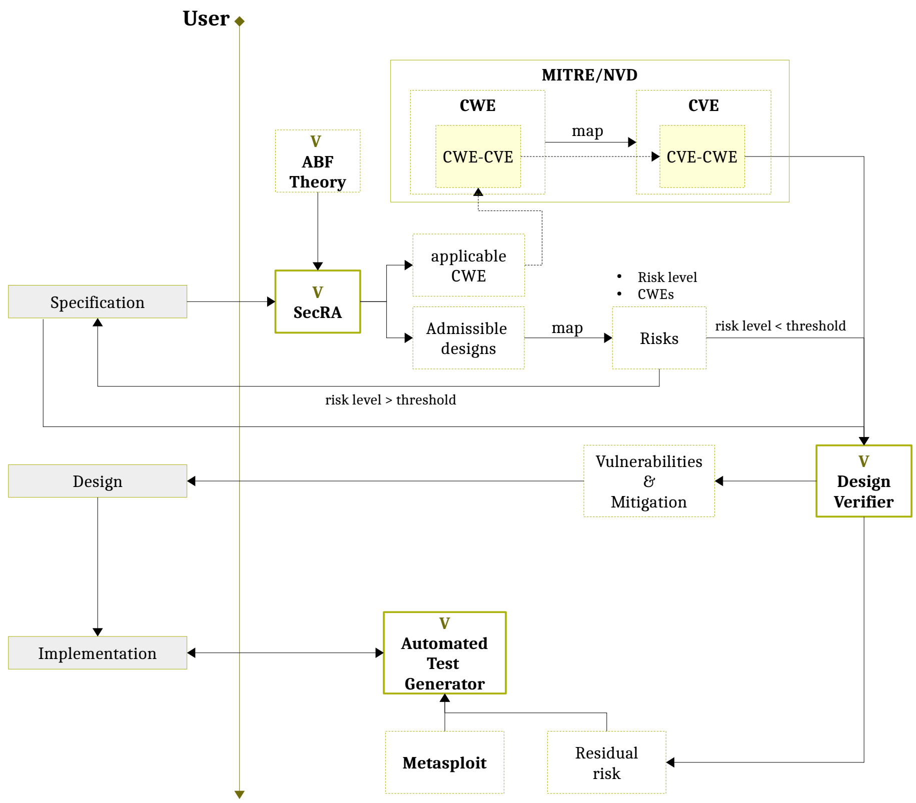


Figure 11: Secure-by-design System Development Life-cycle

In order to explain in details the SPDL, we use the following running example.

Example 4. *As depicted in Figure 12, the use case of the running example (a small part of a subprocess of the SwAT testbed [53]), shows*

- *a tank that is filled with raw water coming from the inlet pipe*
- *a motorized valve (a valve with an actuator) that regulates that*
  - *if opened, allows the raw water to flow into the tank*
  - *if closed, stops the raw water to flow into the tank*
- *a sensor that reads the meters of raw water in the tank*
- *a PLC controller, connected to the sensor and the actuator, that receives the readings from the sensor and regulates the quantity of water in the tank by communicating a change in the state of the actuator based on the following logi*
  - *if the readings from the sensor reports a water level in the tank above (or equal) to 10m, the actuator shall close the valve*

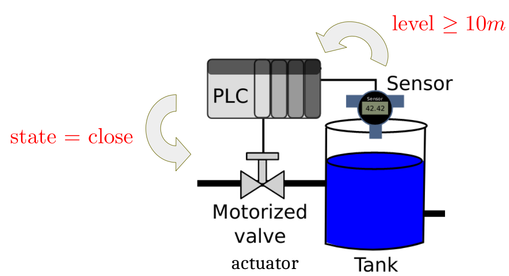


Figure 12: Use Case – Running Example

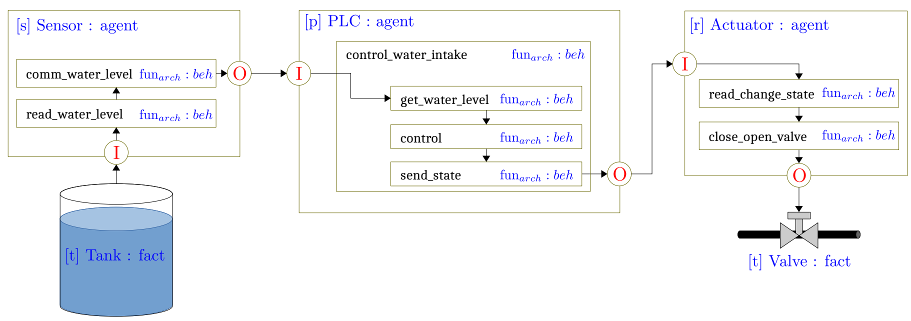


Figure 13: Specification – Running Example

- *if th readings are below 10m the actuator shall open the valve.*

5.1 Specification – Elicitation of Security Requirements

The first step of the SPDL is the definition of the requirements and then, for the purpose of this paper of the (i) security requirements, (ii) architectural (physical and functional) requirements. Our focus in mainly on security requirements but they are necessarily intertwined with the architectural requirements, as we will show afterwards in this section. A complete study on how to properly define the architectural requirements is outside the scope of this paper so we propose a method that is convenient for the purpose of our assessment.

System Requirements. In our running example, there is a system (a CPS) composed by 5 main subsystem which we consider atomic (i.e. not themself composed by subsystems) and then agents: the tank, the sensor, the controller, the actuator, and the motorized valve (valve from now on). The whole specification (which we now describe) of the physical and functional architecture is depicted in Figure 13. In the use case, the communication flow is as follows:

1. the sensor reads the level of water in the tank (tank $\rightarrow$ sensor) [27]

[27] this communication depends on the technology in use for the readings. We assume a unidirectional communication without loss of generality

2. the sensor communicates the readings to the controller (sensor → controller)
3. the controller calculates the valve state and communicates it to the actuator (controller → actuator)
4. the actuator translates (digital to analog) the states received and communicates[28] it to the valve

Each agent has an input or output port from which it communicates with other agents. It is also composed by the functional blocks that defines its functional architecture and physical boundaries that defines its physical architecture, as follows.

req1) the tank has no functional architecture, it is a purely physical component which has the only purpose of containing water

req2) the communication is unidirectional from the tank to the input port of the sensor

req3) the sensor has an input-port that receives only incoming communication from the tank

req4) the sensor has two functional blocks

- req4.1) read_water_level, accepts inputs only from the input-port and outputs a new reading every t seconds
- req4.2) comm_water_level, accepts inputs only from the read_water_level and communicates the readings to the output-port

req5) the sensor has an output-port that only outputs the messages generated by the comm_water_level to the input-port of the controller

req6) the controller has in input-port that only receives incoming communications from the sensor and sends the messages to the functional architecture

req7) the functional architecture of the controller is composed by three functional blocks

- req7.1) get_water_level, accepts inputs only from the input-port and outputs the water level to the main control function
- req7.2) control, calculate the next state of the valve based on the inputs receive from the get_water_level
- req7.3) send_state, receives the output of the control function and communicates it to the output-port

req8) the controller has an output-port that only outputs the messages generated by the send_state to the input-port of the actuator

[28] we do not distinguish, for the sake of simplicity between different types of channels and assume that communicating to the valve produces the expected change of physical state

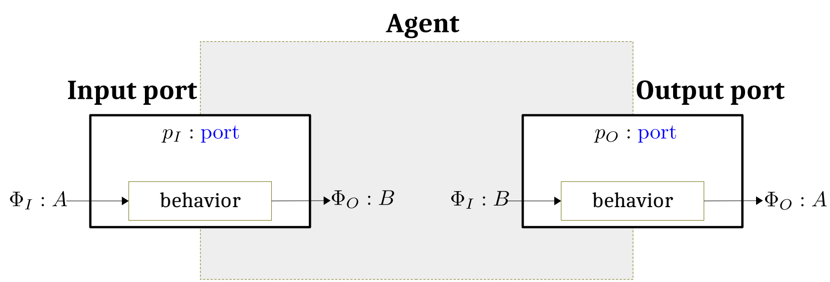


Figure 14: Input and Output Port of an agent

Figure 15: Communication over a Mono-directional Channel

req9) the actuator has an input-port that receives only incoming communication from the controller

req10) the functional architecture of the actuator is composed by two functional blocks

- req10.1) read_change_state, accepts inputs only from the input-port and outputs the next state of the valve to the close_open_valve function
- req10.2) close_open_valve, converts the digital next state received from the read_change_state an analog signal and communicates it the output-port

req11) the valve accepts analog messages from the output port of the actuator and changes its state based on those messages

While *agents* and *communication* between agents are described and defined in Section 4, the concepts of *port* (depicted in Figure 14), *channel* (depicted in Figure 15), *functional and physical architecture*, and *functional block* have not yet been defined with respect to the $\mathcal{ABF}$-theory. Informally, a channel[29] identifies the place where the communication takes place. In this work, we only consider mono-directional channels and communication but the extension to bi-directional channel can be considered as the union of two unidirectional channels. A mono-directional channel is defined by the assertions sent or received (over the channel).

[29] for now, we rely on the intuitive notion of channel as "something" that transfers information as Assertions between output and input port, i.e. mono-directional

Input and Output Ports. Since the $\mathcal{ABF}$-theory is a theory of agents, we consider ports as agents that allows the exchange of information between a channel and another agent. However, a port must be considered as a special type of agent to avoid an *infinite regress* in which a port needs a port to transfer information between the outside of the port to the inside of itself.

Definition 5.1. Input or Output Port – Ports are agents with the following predefined behavior: an input-port changes the type of information from assertion from a sender s to an agent a, $\mathcal{A}_{s \to a}$ to belief of the agent a ($\mathbb{B}_a$) while an output-port from belief to assertion; forwarding information from the outside of an agent's boundary to the inside (input-port) or vice versa (output-port).

The *quality of a port* is determined by the *rcc* relation between the assertions received or sent and the belief, i.e. $rcc(\mathcal{A}_s, \mathbb{B}_a)$ for the input-port or $rcc(\mathcal{A}_r, \mathbb{B}_a)$.

Definition 5.2. Secure Input Port – A secure input port allows information as incoming assertions to flow from a sender s to the behavior of the recipient a (agent). If p_I is a secure input port, $\omega \models \varphi$, and $\varphi \in \mathcal{A}_s$ where s has the only input port p_I, $\omega' \models \mathbb{B}_a$ and $\varphi \in \mathbb{B}_a$.

Port Weaknesses. From this definition, it follows that there exist only the following *six* types of weaknesses, generating six types of insecure port in RCC5 (the notation is reported for input-ports, i.e. on the lef-hand side of the arrow):

W1) *drop port* ($s\boxed{\circ\!\!\rightarrow}a$), where assertions reaches the port but do not pass the boundary of the agent (i.e. do not become belief of the agent)

W2) *insertion port* ($s\boxed{\bullet\!\!\rightarrow}a$), where some new information is believed by a as incoming from the port but s didn't send it

W3) *injection port* ($s\boxed{\circ\!\!\!\bullet\!\!\rightarrow}a$), where information coming from s is substituted with new information which becomes believed by a

W4) *selective port*, where some information passes the port and part is either:

W4.1) dropped ($s\boxed{\circ\!\!\mapsto}a$),

W4.2) inserted ($s\boxed{\bullet\!\!\mapsto}a$), or

W4.3) injected ($s\boxed{\circ\!\!\!\bullet\!\!\mapsto}a$)

Proof. An input port is, in the $\mathcal{ABF}$-theory, defined secure as long as the relation between the two regions of input assertions $\mathcal{A}$ and output beliefs $\mathbb{B}$ are equal, i.e. $EQ(\mathcal{A}, \mathbb{B})$. Therefore, any other relation should result in a weakness (related to an insecurity flaw) of that input port. Using RCC5, there exist exactly other 4 different type of relations, one of which is the discrete-from (DR) relation, i.e. $DR(\mathcal{A}, \mathbb{B})$. When two regions are related by the DR relations, they have no subregion in common. Lets define a function weight $|X|$ such as, for any region X, it represents the smallest possible cardinality of a (mereo)topological base for X; where a base is a collection of regions in a (mereo)topology such that every open region can be written as union of elements of that base. If a communication occurred and results in $DR(\mathcal{A}, \mathbb{B})$, there exist only two mutually exclusive options: either $|\mathcal{A}| = |\mathbb{B}|$ or $|\mathcal{A}| \neq |\mathbb{B}|$.

- If $|\mathcal{A}| \neq |\mathbb{B}|$ (at the end of a communication through the input-port), the two regions have a different number of subregions, and, then, there exist only two mutually exclusive options:
  - either $|\mathcal{A}| > |\mathbb{B}|$, where one ore more Assertions didn't become Beliefs, which we call *drop port* ($s\boxed{\circ\!\!\rightarrow}a$) since its behavior drops and prevents incoming communications,
  - or $|\mathcal{A}| < |\mathbb{B}|$, where there are one ore more Beliefs which have not been sent as input-Assertions, which we call *insertion port* ($s\boxed{\bullet\!\!\rightarrow}a$) since its behavior generates new Beliefs unrelated to the incoming communication.
- If $|\mathcal{A}| = |\mathbb{B}|$, supposing an increasing monotonic entropy on the information (Assertions) exchanged, there exist only two mutually exclusive options:
  - either information has been generated as Belief and transferred as Assertions, increasing the cardinality of both $\mathcal{A}$ and $\mathbb{B}$, i.e. $|\mathcal{A}_{t^0}| > |\mathcal{A}_{t^1}|$ and $|\mathbb{B}_{t^0}| > |\mathbb{B}_{t^1}|$ (where t^0 and t^1 represent time at the beginning of communication and right after, respectively); which means that Assertions have reached the port and new Beliefs have been generated by the port, but no correlation between the elements of the two regions exist. We call this *injection port* ($s\boxed{\LEFTcircle\!\!\rightarrow}a$) since it has modified the information carried by the Assertions into new corresponding Beliefs.
  - or $|\mathcal{A}_{t^0}| = |\mathcal{A}_{t^1}|$ and $|\mathbb{B}_{t^0}| = |\mathbb{B}_{t^1}|$ but this implies that no communication occurred, while we assumed that a communication was taking place (absurd).

In other words, $DR(\mathcal{A}, \mathbb{B}) \equiv \neg O(\mathcal{A}, \mathbb{B}) \equiv \neg(\varphi \subseteq \mathcal{A} \wedge \varphi \subseteq \mathbb{B}) \equiv \varphi \nsubseteq \mathcal{A} \vee \varphi \nsubseteq \mathbb{B}$
Similarly,

- given that $PO(\mathcal{A}, \mathbb{B})$ is equivalent to $DR(\mathcal{A}, \mathbb{B})$ except for the overlapping part (see Appendix A) in which case is $EQ(\mathcal{A}, \mathbb{B})$ (see [71]), we call it a *selective injection port* ($s\boxed{\LEFTcircle\!\!\mapsto}a$); otherwise
- $PP(\mathcal{A}, \mathbb{B}) \equiv \mathcal{A} \Leftarrow \mathbb{B}$ and $PPi(\mathcal{A}, \mathbb{B}) \equiv \mathcal{A} \Rightarrow \mathbb{B}$ are subcases of $EQ(\mathcal{A}, \mathbb{B}) \equiv \mathcal{A} \Leftrightarrow \mathbb{B}$, and then
  - if $PP(\mathcal{A}, \mathbb{B})$, we have that some Beliefs are not correlated to incoming Assertions and we call it *selective insertion port* ($s\boxed{\bullet\!\!\mapsto}a$),
  - if $PPi(\mathcal{A}, \mathbb{B})$ we have that some incoming Assertions are not correlated to Belief and we call it *selective drop port* ($s\boxed{\circ\!\!\mapsto}a$)

□

Communication Channels. We start by considering the difference between a (communication) mono-directional channel (channel from now on) and an agent, as we did for the ports, since the $\mathcal{ABF}$-theory is a theory of agents. In fact, if a channel were considered an agent (channel-agent) then the question would be how an agent would transfer its Assertions to the channel-agent. If the channel between the agent and the channel-agent is again an agent, we would generate an *infinite regress*. Therefore, we do allow channel-agents but we assume a finite depth (of detail) for a channel, where there exists a bottom-channel which is not an agent. For now, we do not constrain a channel-agent in any way so there is no difference between a channel agent and agent. Therefore, for the sake of simplicity, we consider, in this text, channels to be bottom-channels, defined as agents with the pre-defined behavior (i.e. defined in a dogmatic way) of forwarding any input-assertion as output-assertion, without modifying it[30].

Definition 5.3. Secure Mono-directional Channel (bottom-channel) – A mono-directional channel between the two agents ($s \rightarrow r$) is defined as an agent which behavior is (dogmatically) defined as: to forward any Assertion received from s over an input-port, to the output-port where r is listening to.

The *quality of a mono-directional* channel is defined as the rcc relation between the Assertions made by the sender and the ones received by the receiver, i.e. $rcc(\mathcal{A}_s, \mathcal{A}_r)$.

Channel Weaknesses. Given that a mono-directional bottom-channel is assumed to be perfectly forwarding any Assertion from its input-port to its output-port, there is no insecure behavior but only the combination of the weaknesses of the input and output port; therefore there exists $(7^2) - 2 = 47$ theoretical configurations (7^2 because there are 6 insecure types of port, plus 1 secure type, on both input and oupput side; and we exclude the configuration with 2 secure types as input and ouput, -2); where only 43 are possible.

W5) *secure input port* and *output drop port* ($s \rightarrow\!\circ\ r$),

W6) *secure input port* and *output insertion port* ($s \rightarrow\!\bullet\ r$),

W7) *secure input port* and *output injection port* ($s \rightarrow\!◐\ r$),

W8) *secure input port* and *output selective drop port* ($s \rightarrow\!\shortmid\!\circ\ r$),

W9) *secure input port* and *output selective insertion port* ($s \rightarrow\!\shortmid\!\bullet\ r$),

W10) *secure input port* and *output selective injection port* ($s \rightarrow\!\shortmid\!◐\ r$),

W11) *input drop port* and *output drop port* ($s \circ\!\!\rightarrow\!\circ\ r$)

W12) *input drop port* and *output insertion port* ($s \circ\!\!\rightarrow\!\bullet\ r$)

W13) *input drop port* and *output secure port* ($s \circ\!\!\rightarrow r$)

30 Nothing prevents us from introducing additional constraints to the channel as storing Assertions that are transferred over the channel, or filter out some input-assertions.

W14) *input insert port* and *output drop port* ($s \bullet\!\!\rightarrow\!\circ\ r$)

W15) *input insert port* and *output insert port* ($s \bullet\!\!\rightarrow\!\bullet\ r$)

W16) *input insert port* and *output injection port* ($s \bullet\!\!\rightarrow\!\LEFTcircle\ r$)

W17) *input insert port* and *output selective drop port* ($s \bullet\!\!\rightarrow\!|\circ\ r$)

W18) *input insert port* and *output selective insertion port* ($s \bullet\!\!\rightarrow\!|\bullet\ r$)

W19) *input insert port* and *output selective injection port* ($s \bullet\!\!\rightarrow\!|\LEFTcircle\ r$)

W20) *input insert port* and *output secure port* ($s \bullet\!\!\rightarrow r$)

W21) *input injection port* and *output drop port* ($s \LEFTcircle\!\!\rightarrow\!\circ\ r$)

W22) *input injection port* and *output insertion port* ($s \LEFTcircle\!\!\rightarrow\!\bullet\ r$)

W23) *input injection port* and *output injection port* ($s \LEFTcircle\!\!\rightarrow\!\LEFTcircle\ r$)

W24) *input injection port* and *output selective drop port* ($s \LEFTcircle\!\!\rightarrow\!|\circ\ r$)

W25) *input injection port* and *output selective insertion port* ($s \LEFTcircle\!\!\rightarrow\!|\bullet\ r$)

W26) *input injection port* and *output selective injection port* ($s \LEFTcircle\!\!\rightarrow\!|\LEFTcircle\ r$)

W27) *input injection port* and *output secure port* ($s \LEFTcircle\!\!\rightarrow r$)

W28) *input selective drop port* and *output drop port* ($s \circ\!\!\mapsto\!\circ\ r$)

W29) *input selective drop port* and *output insertion port* ($s \circ\!\!\mapsto\!\bullet\ r$)

W30) *input selective drop port* and *output injection port* ($s \circ\!\!\mapsto\!\LEFTcircle\ r$)

W31) *input selective drop port* and *output selective drop port* ($s \circ\!\!\mapsto\!|\circ\ r$)

W32) *input selective drop port* and *output selective insertion port* ($s \circ\!\!\mapsto\!|\bullet\ r$)

W33) *input selective drop port* and *output selective injection port* ($s \circ\!\!\mapsto\!|\LEFTcircle\ r$)

W34) *input selective drop port* and *output secure port* ($s \circ\!\!\mapsto r$)

W35) *input selective insertion port* and *output drop port* ($s \bullet\!\!\mapsto\!\circ\ r$)

W36) *input selective insertion port* and *output insertion port* ($s \bullet\!\!\mapsto\!\bullet\ r$)

W37) *input selective insertion port* and *output injection port* ($s \bullet\!\!\mapsto\!\LEFTcircle\ r$)

W38) *input selective insertion port* and *output selective drop port* ($s \bullet\!\!\mapsto\!|\circ\ r$)

W39) *input selective insertion port* and *output selective insertion port* ($s \bullet\!\!\mapsto\!|\bullet\ r$)

W40) *input selective insertion port* and *output selective injection port* ($s \bullet\!\!\mapsto\!|\LEFTcircle\ r$)

W41) *input selective insertion port* and *output secure port* ($s \bullet\!\!\mapsto r$)

W42) *input selective injection port* and *output drop port* ($s \LEFTcircle\!\!\mapsto\circ\, r$)

W43) *input selective injection port* and *output insertion port* ($s \LEFTcircle\!\!\mapsto\bullet\, r$)

W44) *input selective injection port* and *output injection port* ($s \LEFTcircle\!\!\mapsto\LEFTcircle\, r$)

W45) *input selective injection port* and *output selective drop port* ($s \LEFTcircle\!\!\mapsto\!\!\mid\!\circ\, r$)

W46) *input selective injection port* and *output selective insertion port* ($s \LEFTcircle\!\!\mapsto\!\!\mid\!\bullet\, r$)

W47) *input selective injection port* and *output selective injection port* ($s \LEFTcircle\!\!\mapsto\!\!\mid\!\LEFTcircle\, r$)

W48) *input selective injection port* and *output secure port* ($s \rightarrow\!\LEFTcircle\, r$)

In Appendix B, we report the proof which is exhaustive on all possible combinations.

Functional Block. A Functional Architecture, as already described beforehand, takes Information as input-Beliefs and transforms the Information into output-Beliefs. Those transformations occurs within the Functional Architecture, where Functional Blocks transforms Beliefs into other Beliefs. Similarly to channels, we could consider a Functional Block as a Functional Architecture occurring in an infinite regress. Therefore, we consider Functional Blocks as correctly executing their abstract behavior, correctly transforming inputs into outputs (Beliefs).

Definition 5.4. Functional Block and Architecture – A functional block of an agent a takes a region of input-Beliefs and outputs a region of output-Beliefs. A functional architecture is an interconnected system of functional blocks.

The quality of a functional block or architecture cannot be expressed as a relation between input and output belief since there is a transformation process in the functional block which will be defined during the design and implementation phase, but not necessarily known in details in the specification phase. In order to define the quality of functional block we first need to introduce the engineering concept of Facts.

Facts. During the specification phase, for any agent, channel, port, functional block and architecture, there exist a Fact predicating over them. In other words, any requirement is defined as a Fact since they must be true in any design or implementation. As we stated in Section 4 and depicted in Figure 10, Facts are strategic rules that defines how the system shall behave (by specification), while reality may be shown to be insecure.

The quality of a Functional Block is determined by the relation between the input-Beliefs formulas and the symmetric formulas in the Fact region (equivalently for output-Beliefs). Therefore, the quality of the functional architecture is not a purely architectural property but relies on the logic implemented in the functional blocks (which we abstract away at this stage of the engineering model).

Definition 5.5. Secure Functional Block and Architecture – A functional block is secure if the input beliefs and output beliefs are in EQ relation with the corresponding Facts mandated by requirements. A functional architecture composed exclusively by secure functional blocks is considered secure.

Therefore, there exists the following weaknesses in the functional architecture, where $\mathbb{B}_I$ and $\mathbb{B}_O$ are input and output Beliefs respectively, and $\mathcal{F}_I$ and $\mathcal{F}_O$ are the corresponding Facts (i.e. what really happens).

1. $DR(\mathbb{B}_I, \mathcal{F}_I)$ and $DR(\mathbb{B}_O, \mathcal{F}_O)$, the input beliefs are misinterpreted and this result in the complete alteration of the output beliefs
2. **tofix:** TBD

Security Requirements. While standards (such as IEC62443-1-3**FIX**[31]) defines requirements as "information in transit/at-rest should be encrypted" we believe that, at specification level, security requirements can be formulated as subcategories of the well-known CIA triad (where CIA stands for Confidentiality, Integrity, Availability). Infosec, or Information Security, defines the security risks related to information with the de-facto standard CIA triad, which is often criticized [**CIAcriticismCPS**] for being too general or non-adequate (e.g. by adding authenticity which is often used as a building block for confidentiality and integrity) to be effectively applied to the engineering of systems. Due to this, many researchers and organization tried to improve the CIA triad. The evolutions/extensions of the CIA triad has been documented, e.g., in [70] and is summarized in Table 3.

Year	Definition	Legend
1970s	infosec = CIA	Confidentiality, Integrity, Availability
1980s	infosec += (Au, nR)	Authenticity and non-Repudiation
1990s	infosec += CSpec	Correctness in Specification
2000s	infosec += RITE	Responsibility, Integrity of people, Trust, Ethicality

Table 3: Chronological progression of the CIA triad

An representative example is the effort made by the OECD (Organisation for Economic Co-operation and Development) in [28] to define security guidelines for information system (1992) based on new principles such as "awareness, ethics, risk assessment" and maintain those revising the document (e.g. following the multistakeholder expert consultation in 2013). Other similar efforts focused on defining entirely new principles or extending the CIA triad, such as [59]. What is of interest for our argument is that, to the best of our knowledge, all of the approaches that aim at improving the CIA triad are besed on motivations related to empirical evidences. On the contrary, we now define (in Section 5.1) the CIA triad for an agent defined in the $\mathcal{ABF}$-theory and we test if (i) other properties are allowed in the $\mathcal{ABF}$-theory, and (ii) if and how the CIA triad can be detailed in the $\mathcal{ABF}$-theory.

[31] **mr:** check the correct version

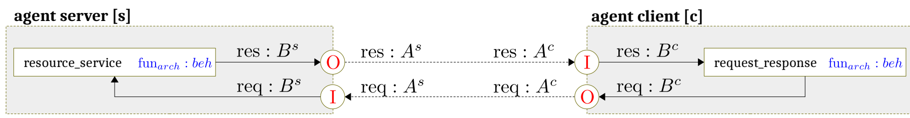


Figure 16: Client-Server paradigm in $\mathcal{ABF}$-theory

In [2, 70], CIA are defined as follows.

- Unauthorized information release: an unauthorized person is able to read and take advantage of information stored in the computer. This category of concern sometimes extends to "traffic analysis," in which the intruder only observes the patterns of information use. From those patterns, the intruder can infer some information content. This category also includes the unauthorized use of a proprietary program. (Confidentiality or Secrecy)
- Unauthorized information modification: an unauthorized person is able to make changes in stored information – a form of sabotage. It should be noted that in the case of this kind of violation, the intruder does not necessarily see the information he has changed. (Integrity)
- Unauthorized denial of use: an intruder can prevent an authorized user from referring to, or from modifying information, even though the intruder may not be able to refer to, neither modify the information themselves. (Availability)

We now want to generalize the CIA triad to agents and not just information and map it to the $\mathcal{ABF}$-theory. For the sake of simplicity, we revert the order and start from availability.

5.1.1 Availability and Integrity (Architecture Security)

Availability, again, is the denial of use of a resource. In the $\mathcal{ABF}$-theory, in order to prevent a resource to be available, there exists only (i.e. inclusively) the following mutually exclusive cases that we illustrate using a client-server paradigm (depicted in Figure 16).

1. The resource on the server cannot be found, so that it is possible for the client to contact the server and the server sends a response back, but the output-port does not receive the necessary information. This may be due to a software error such that the functional architecture doesn't output a response, or a problem in the flow of information (i.e. the functional architecture doesn't receive the necessary information from other resources), or an hardware failure &c. This is formalized as the relation between the beliefs produced by the functional architecture and the ones received by the output port, i.e. $rcc(\mathcal{B}_s, \{\text{res}\})$ or $rcc(\mathbb{B}_s, \{\text{res}\})$

2. The server hosting the resource is down, and then the server cannot output responses (i.e. insecure output port due to drop-related issue) or receive requests (insecure input port due to drop-related issue) or both.
3. The client cannot properly compute request (simmetrically to Case 1)
4. The client is down, symmetric to Case 2
5. The channel is down due to a drop related issue in one of the two channels

We shall, then, distinguish between two cases:

Definition 5.6. Agent-Availability – when the property of availability is expressed over an agent or a system, the list of possible designs violating the availability property can be shown as the designs that allow drop ports or drop channels.

FIX[32]

Definition 5.7. Infosec-Availability – when the property of availability is expressed over a formula, the list of possible designs that violates the availability property is reduced to the part of the system that influences that specific variable.

In a similar way we can define integrity since this requirement may be expressed over an agent/system or a specific formula (i.e. element of a Region).

Definition 5.8. Agent-Integrity – when the property of integrity is expressed over an agent or a system, the list of possible designs violating the integrity property can be shown as the designs that allow insert or inject ports, or inert or inject channels.

Definition 5.9. Infosec-Integrity – when the property of integrity is expressed over a formula, the list of possible designs that violates the integrity property is reduced to the part of the system that influences that specific variable.

While availability and integrity are architectural property, meaning that they can be expressed over the behavior of a port or a channel (as constituent of the physical or functional architecture), confidentiality is not. Confidentiality states that an information shall not be understood and not that the information shall not be retrieved. So, if a port leaks an entire message, there is no confidentiality issue in general, unless that message was not encrypted at all, in which case the confidentiality leakage is straightforward. On the other hand, if the information is not provided or altered there is an availability or integrity issue. A similar argument can be proposed for integrity sub-properties such as authenticity or non-repudiation. No wonder why it is often stated that availability and integrity

[32] **mr:** all security definitions defines when the property is not preserved instead of when it's preserved

are the first properties to be considered in CPS while confidentiality takes the first place when dealing with security protocols. In fact, integrity is often assumed by pattern-matching (e.g. in AVANTSSAR [**avantssar**]) and availability is hardly ever considered in protocol verification. The question is, can we express properties such as confidentiality, authentication, or non-repudiation as architectural properties in $\mathcal{ABF}$-theory? Confidentiality (and Availability) is based on the concept of secrecy so we shall first start from describing what secrecy is and how it can be used to guarantee confidentiality in a system.

5.1.2 Secrecy, Authenticity, and Key Agreement (Cryptographic Security)

As Maurer puts it ([54]), there exist two types of cryptographic security (i.e. scientific theories aiming at guaranteeing security through the use of cryptographic functions): " The security of a cryptographic system can rely either on the computational infeasibility of breaking it (computational security), or on the theoretical impossibility of breaking it, even using in infinite computing power (information-theoretic or unconditional security)". We are not currently interested in detailing the difference between the two and we simplify it as follows. A computationally-secure cryptographic function (usually called encryption function) $E(\varphi) = e$ (where φ is a plain-text and e a cipher-text) can be considered perfectly secure if a malicious agent that obtains the result of the cryptographic function (e) has bounded computational power, while a unconditional secure cryptographic functions are considered perfectly secure. In this context, security means that is is impossible to calculate φ from e, therefore impossible to compute $D(e) = \varphi$. As one can evince from [**Shannon1948information**], the commonality between the two types of cryptographic functions is that they both rely on the assumption about the probabilistic behavior of the universe. They do so, for the random (i.e. with high entropy [**Shannon1948information**]) generation of the so called *key*, a piece of information, parametric to the cryptographic function, used to guarantee the security. Therefore, we can conclude that it is feasible for an agent who knows k to decrypt the cipher-text obtaining the clear-text (i.e. $D(k, E(k, \varphi)) = \varphi$) while it is impossible for any agent who has not the proper key. In this case, the information is considered to be *secret or confidential*.

Definition 5.10. Infosec-Secrecy or Confidentiality – When an Assertion has been generated as an output of a functional or physical agent, port, or functional block such that it is said (in a factorial way) to encrypt the input, that Assertion is considered confidential.

The quality of the confidentiality of information (Assertions or Beliefs depending on the subsystem considered) are inversely proportional to the number of integrity issues of the subsystem.

tofix: Authenticity: assurance that the information is from the source it claims to be from. Authenticity involves proof of identity. Authenticity is verified through authentication. If information X arrives from A to B, A sent information X and B can verify it. ??Authenticity implies integrity but Integrity doesn't imply authenticity?? ??Authenticity doesn't imply confidentiality but confidentiality does imply authenticity??

Non-repudiation: the ability to ensure that a party to a contract or a communication cannot deny the authenticity of their signature on information (or the sending of) that they originated. If information X arrives from A, A sent it and cannot deny it (partial overlap??). If sender A sends information X, it cannot say he didn't do it. Non-repudiation implies authenticity and integrity. Authenticity doesn't imply non-repudiation (replay attacks?). Integrity doesn't imply non-repudiation (replay attacks?).

5.2 Abstraction Level of a Specification in $\mathcal{ABF}$-theory

As highlighted in [], "The level of abstraction is always a concern when it come sto the definition of a system model." (as we already mentioned in Section 2).

6 A Prototype for Security Risk Assessment

7 Conclusion

In this paper, we proposed a foundational theory on security, arguing that security-related issues are not related to the maliciousness of an agent but to the vagueness of the security controls on the engineering processes. The current state-of-the-art processes allows, given a specification, an engineer to design the system in such a way that security issues arise due to the lack of proper security risk assessment processes. Those design, again, lack of security verification processes based on a solid security foundational theory and then permit the generation of insecure implementation. The security verification and test-case generation will be the focus of our next steps.

We may conclude that the problem of security is a problem related to the many possible design and, in turn, implementation given a specification. Philosophically, the problem is similar to the epistemological search for truth, where the challenge is to relate Information and Belief to Human Knowledge. Scientifically, generalizing the Human to an Agent, the problem is to relate Assertions and Behaviors, to Facts. From an engineering standpoint, by generalizing the concept of Agents as architectural subsystems, the problem is to link Channels (and Ports) and Functional Architectures to Requirements.

A Selective Injection Port

$$PO(x,y) = O(x,y) \wedge \neg P(x,y) \wedge \neg P(y,x)$$

and

$$\begin{aligned} O(x,y) = \exists z.\ P(z,x) & \qquad\qquad \wedge P(z,y) \\ \forall z'.\ z' \subseteq z \Rightarrow z' \subseteq x & \qquad\qquad \forall z''.\ z'' \subseteq z \Rightarrow z' \subseteq y \end{aligned}$$

and

$$\begin{aligned} \neg P(x,y) =& \neg[\forall \varphi.\ (\varphi \subseteq x) \Rightarrow (\varphi \subseteq y)] \\ & \neg[\forall \varphi.\ \neg(\varphi \subseteq x) \vee (\varphi \subseteq y)] \\ & \exists \varphi.\ (\varphi \subseteq x) \wedge (\varphi \not\subseteq y) \end{aligned}$$

Therefore

$$PO(\mathcal{A}, \mathbb{B}) = (\varphi \subseteq \mathcal{A} \wedge \varphi \subseteq \mathbb{B}) \wedge (\varphi' \subseteq \mathcal{A} \wedge \varphi' \not\subseteq \mathbb{B}) \wedge (\varphi'' \subseteq \mathcal{A} \wedge \varphi'' \subseteq \mathbb{B})$$

Which means that, when a communication over an input-port results in $PO(\mathcal{A}, \mathbb{B})$, the port is:

- correctly receiving incoming communication as Assertion and sending it to the recipient agent as Beliefs, *and* $(s \rightarrow a)$
- incorrectly, dropping some input communication (Assertions) $(s \boxed{\circ\!\!\rightarrow} a)$, *and*
- incorrectly, inserting some new Beliefs $(s \boxed{\bullet\!\!\rightarrow} a)$

Which we abbreviate as a *selective injection* port: $s \boxed{\odot\!\!\mapsto} a$.

B Insecure Communication Port

tofix: this proof needs to be done

Proof. In Appendix **??**, we report the proof which is exhaustive on all possible combinations. In the following we report only the excluded cases and assume the other to be strightforwad (as a proof sketch).

1. *input secure port* and
   (a) *output drop port* $(s \rightarrow\!\circ\ r)$,
   (b) *output insertion port* $(s \rightarrow\!\bullet\ r)$,
   (c) *output injection port* $(s \rightarrow\!\odot\ r)$,
   (d) *output selective drop port* $(s \nrightarrow\!\circ\ r)$,
   (e) *output selective insertion port* $(s \nrightarrow\!\bullet\ r)$,
   (f) *output selective injection port* $(s \nrightarrow\!\odot\ r)$,

2. *input drop port* (all the input-assertions are dropped) and

   (a) *output drop port* ($s \circ\!\!\rightarrow\!\circ\ r$), this configuration is possible but the input port doesn't forward any message therefore the output port will never have a message to drop

   (b) *output insertion port* ($s \circ\!\!\rightarrow\!\bullet\ r$), where all the ouput-assertions have been inserted by the output port

   (c) *output injection port*, this configuration is impossible since there is no Assertions from the input-port to inject (or it behaves as $s \circ\!\!\rightarrow\!\bullet\ r$)

   (d) *output selective drop port*, as in 2c there is no Assertion from the input port therfore no selective port is possible in general. In this case this configuration is impossible or behaves as $s \circ\!\!\rightarrow\!\circ\ r$

   (e) *output selective insertion port*, as in 2d impossible, or behaves as $s \circ\!\!\rightarrow\!\bullet\ r$

   (f) *output selective injection port*, as in 2d impossible, or behaves as $s \circ\!\!\rightarrow\!\bullet\ r$

   (g) *ouput secure port* ($s \circ\!\!\rightarrow r$)

3. *input insertion port* (where all the assertion reaching the output-port have been artificially generated by the input-port) and

   (a) *output drop port* ($s \bullet\!\!\rightarrow\!\circ\ r$), this configuration is possible but the drop port drops every inserted message.

   (b) *output insertion port* ($s \bullet\!\!\rightarrow\!\bullet\ r$), new Assertions have all been inserted both at the left-hand side and righ-hand side of the communication.

   (c) *output injection port* ($s \bullet\!\!\rightarrow\!\LEFTcircle\ r$), the new Assertions, inserted in the left-hand side, are then modified/injected on the right-hand side of the communication

   (d) *output selective drop port* ($s \bullet\!\!\rightarrow\!\mid\!\circ\ r$), where some insertions of the input port are dropped

   (e) *output selective insertion port* ($s \bullet\!\!\rightarrow\!\mid\!\bullet\ r$), where some insertions of the input port are substituted with new insertion on the right-hand side of the communication

   (f) *output selective injection port* ($s \bullet\!\!\rightarrow\!\mid\!\LEFTcircle\ r$) where some insertions of the input port are modified/injected on the right-hand side of the communication

   (g) *output secure port* ($s \bullet\!\!\rightarrow r$)

4. *input injection port* and

   (a) *output drop port* ($s \LEFTcircle\!\!\rightarrow\!\circ\ r$)

   (b) *output insertion port* ($s \LEFTcircle\!\!\rightarrow\!\bullet\ r$)

   (c) *output injection port* ($s \LEFTcircle\!\!\rightarrow\!\LEFTcircle\ r$)

(d) *output selective drop port* ($s \LEFTcircle\!\!\rightarrow\!\!| \circ\ r$)

(e) *output selective insertion port* ($s \LEFTcircle\!\!\rightarrow\!\!| \bullet\ r$)

(f) *output selective injection port* ($s \LEFTcircle\!\!\rightarrow\!\!| \LEFTcircle\ r$)

(g) *output secure port* ($s \LEFTcircle\!\!\rightarrow r$)

5. *input selective drop port* and

(a) *output drop port* ($s \circ\!\!\mapsto\!\circ\ r$)

(b) *output insertion port* ($s \circ\!\!\mapsto\!\bullet\ r$)

(c) *output injection port* ($s \circ\!\!\mapsto\!\LEFTcircle\ r$)

(d) *output selective drop port* ($s \circ\!\!\mapsto\!\!| \circ\ r$)

(e) *output selective insertion port* ($s \circ\!\!\mapsto\!\!| \bullet\ r$)

(f) *output selective injection port* ($s \circ\!\!\mapsto\!\!| \LEFTcircle\ r$)

(g) *output secure port* ($s \circ\!\!\mapsto r$)

6. *input selective insertion port* and

(a) *output drop port* ($s \bullet\!\!\mapsto\!\circ\ r$)

(b) *output insertion port* ($s \bullet\!\!\mapsto\!\bullet\ r$)

(c) *output injection port* ($s \bullet\!\!\mapsto\!\LEFTcircle\ r$)

(d) *output selective drop port* ($s \bullet\!\!\mapsto\!\!| \circ\ r$)

(e) *output selective insertion port* ($s \bullet\!\!\mapsto\!\!| \bullet\ r$)

(f) *output selective injection port* ($s \bullet\!\!\mapsto\!\!| \LEFTcircle\ r$)

(g) *output secure port* ($s \bullet\!\!\mapsto r$)

7. *input selective injection port* and

(a) *output drop port* ($s \LEFTcircle\!\!\mapsto\!\circ\ r$)

(b) *output insertion port* ($s \LEFTcircle\!\!\mapsto\!\bullet\ r$)

(c) *output injection port* ($s \LEFTcircle\!\!\mapsto\!\LEFTcircle\ r$)

(d) *output selective drop port* ($s \LEFTcircle\!\!\mapsto\!\!| \circ\ r$)

(e) *output selective insertion port* ($s \LEFTcircle\!\!\mapsto\!\!| \bullet\ r$)

(f) *output selective injection port* ($s \LEFTcircle\!\!\mapsto\!\!| \LEFTcircle\ r$)

(g) *output secure port* ($s \rightarrow\!\LEFTcircle\ r$)

□

References

[1] International Electrotechnical Commission (IEC). *IEC 61511-1:2016+AMD1:2017 CSV Consolidated version*. Aug. 16, 2017. URL: https://webstore.iec.ch/publication/61289 (visited on 01/21/2020).

[2] James Anderson. *Computer Security Technology Planning Study*. Oct. 1972. URL: https://csrc.nist.gov/csrc/media/publications/conference-paper/1998/10/08/proceedings-of-the-21st-nissc-1998/documents/early-cs-papers/ande72.pdf.

[3] Alessandro Armando, Roberto Carbone, and Luca Compagna. "SATMC: a SAT-based model checker for security protocols, business processes, and security APIs". In: *International Journal on Software Tools for Technology Transfer* 18.2 (2016), pp. 187–204.

[4] David Basin, Sebastian Mödersheim, and Luca Vigano. "OFMC: A symbolic model checker for security protocols". In: *International Journal of Information Security* 4.3 (2005), pp. 181–208.

[5] Rebecca M. Blank and Patrick D. Gallagher. "NIST Special Publication 800-53 Revision 4 - Security and Privacy Controls for Federal Information Systems and Organizations". In: *National Institute of Standards and Technology Special Publication* (Apr. 2013). URL: http://dx.doi.org/10.6028/NIST.SP.800-53r4.

[6] Blanchet Bruno. "Symbolic and Computational Mechanized Verification of the ARINC823 Avionic Protocols". In: *Computer Security Foundations Symposium (CSF)*. IEEE, Aug. 2017, pp. 68–82.

[7] Tom Caddy. "Penetration Testing". In: *Encyclopedia of Cryptography and Security*. Ed. by Henk C. A. van Tilborg. Boston, MA: Springer US, 2005, pp. 456–457. ISBN: 978-0-387-23483-0. DOI: 10.1007/0-387-23483-7_297. URL: https://doi.org/10.1007/0-387-23483-7_297.

[8] *CAPEC-104*. Sept. 30, 2019. URL: https://capec.mitre.org/data/definitions/104.html (visited on 02/19/2020).

[9] *CAPEC-73*. Sept. 30, 2019. URL: https://capec.mitre.org/data/definitions/73.html (visited on 02/19/2020).

[10] *CAPEC-81*. Sept. 30, 2019. URL: https://capec.mitre.org/data/definitions/81.html (visited on 02/19/2020).

[11] *CAPEC-85*. Sept. 30, 2019. URL: https://capec.mitre.org/data/definitions/85.html (visited on 02/19/2020).

[12] *Common Attack Pattern Enumeration and Classification*. URL: https://capec.mitre.org/ (visited on 02/19/2020).

[13] *Common Weakness Enumeration (CWE)*. Jan. 9, 2019. URL: https://cwe.mitre.org/ (visited on 02/04/2020).

[14] The MITRE Corporation. *CVE – Common Vulnerabilities and Exposures*. Jan. 16, 2020. URL: https://cve.mitre.org/ (visited on 01/22/2020).

[15] *CVE-2008-0005*. Dec. 3, 2007. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-0005 (visited on 02/04/2020).

[16] *CVE-2008-0757*. Feb. 13, 2008. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-0757 (visited on 02/04/2020).

[17] *CVE-2008-0769*. Feb. 13, 2008. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-0769 (visited on 02/04/2020).

[18] *CVE-2008-3773*. Aug. 22, 2008. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-3773 (visited on 02/04/2020).

[19] *CVE-2008-4636*. Oct. 21, 2018. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-4636 (visited on 02/04/2020).

[20] *CVE-2008-5573*. Dec. 15, 2008. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-5573 (visited on 02/04/2020).

[21] *CWE VIEW: Research Concepts*. 2020. URL: https://cwe.mitre.org/data/definitions/1000.html (visited on 02/03/2020).

[22] *Cybersecurity: Industry Report & Investment Case*. June 25, 2018. URL: https://www.nasdaq.com/articles/cybersecurity-industry-report-investment-case-2018-06-25 (visited on 01/22/2020).

[23] *Dialectical Materialism*. Progress Publishers, 1983. URL: https://www.marxists.org/reference/archive/spirkin/works/dialectical-materialism/index.html.

[24] *Digital Single Market – Policy – Cybersecurity industry*. Sept. 30, 2019. URL: https://ec.europa.eu/digital-single-market/en/cybersecurity-industry (visited on 01/22/2020).

[25] *Digital Single Market – Policy – Standards*. Dec. 6, 2019. URL: https://ec.europa.eu/digital-single-market/en/standards-digitising-european-industry (visited on 01/22/2020).

[26] *Digital Single Market – Press Release – Commission signs agreement with cybersecurity industry to increase measures to address cyber threats*. July 5, 2016. URL: https://ec.europa.eu/digital-single-market/en/news/commission-signs-agreement-cybersecurity-industry-increase-measures-address-cyber-threats (visited on 01/22/2020).

[27] Danny Dolev and Andrew Yao. "On the security of public key protocols". In: *IEEE Transactions on information theory* 29.2 (1983), pp. 198–208.

[28] OECD (Organisation for Economic Co-operation and Development). *Guidelines for the Security of Information Systems and Networks*. 2013. URL: https://www.sbs.ox.ac.uk/cybersecurity-capacity/content/oecd-guidelines-security-information-systems-and-networks.

[29] Sextus Empiricus. *Outline of Pyrrhonism*. Prometheus Book, 1990. ISBN: 978-0-87975-597-3.

[30] Institution of Engineering and Technology (IET). *Glossary of safety terminology*. Jan. 2017. URL: https://www.theiet.org/media/1435/hsb00.pdf.

[31] Santiago Escobar, Catherine A. Meadows, and José Meseguer. "Maude-NPA: Cryptographic Protocol Analysis Modulo Equational Properties". In: *Foundations of Security Analysis and Design (FOSAD) - Tutorial Lectures*. 2007, pp. 1–50.

[32] *FAQ – What is the difference between a software vulnerability and software weakness?* Sept. 17, 2019. URL: https://cwe.mitre.org/about/faq.html#A.2 (visited on 02/03/2020).

[33] James Garson. *Modal Logic*. Ed. by Edward N. Zalta. 2018.

[34] Rolf Grütter, Thomas Scharrenbach, and Bettina Bauer-Messmer. "Improving an RCC-Derived Geospatial Approximation by OWL Axioms". In: *ISWC*. 2008.

[35] Rolf Grütter, Thomas Scharrenbach, and Bettina Bauer-Messmer. "Improving an RCC-Derived Geospatial Approximation by OWL Axioms". In: *The Semantic Web (ISWC)*. Ed. by Amit Sheth et al. Springer Berlin Heidelberg, 2008, pp. 293–306.

[36] Albert B. Hakim. *Historical Introduction to Philosophy*. Routledge, 2016. ISBN: 978-0-13-190005-9.

[37] Cormac Herley. "Justifying Security Measures—a Position Paper". In: *European Symposium on Research in Computer Security*. Springer. 2017, pp. 11–17.

[38] Cormac Herley. "So long, and no thanks for the externalities: the rational rejection of security advice by users". In: *Proceedings of the 2009 workshop on New security paradigms workshop*. 2009, pp. 133–144.

[39] Cormac Herley. *The Unfalsifiability of Security Claims - Invited Talk USENIX Security*. Aug. 2016. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/herley (visited on 01/15/2020).

[40] Cormac Herley. "Unfalsifiability of security claims". In: *Proceedings of the National Academy of Sciences (PNAS)* 113.23 (2016), pp. 6415–6420.

[41] Jaakko Hintikka. "Knowledge and Belief: An Introduction to the Logic of the Two Notions". In: *Studia Logica* 16 (1962), pp. 119–122.

[42] Jakko Hintikka. "On proper (popper?) and improper uses of information in epistemology". In: *Theoria*. Wiley Online Library, 1993, pp. 158–165. URL: https://doi.org/10.1111/j.1755-2567.1993.tb00867.x.

[43] *Improper Encoding or Escaping of Output*. June 20, 2019. URL: https://cwe.mitre.org/data/definitions/116.html (visited on 02/04/2020).

[44] Merriam-Webster Inc. *failure*. 2020. URL: https://www.merriam-webster.com/dictionary/failure (visited on 01/18/2020).

[45] Merriam-Webster Inc. *Realization*. 2020. URL: https://www.merriam-webster.com/dictionary/realization (visited on 02/19/2020).

[46] Wikipedia Foundation Inc. *Belief–desire–intention software model*. Dec. 22, 2019. URL: https://en.wikipedia.org/wiki/Belief%E2%80%93desire%E2%80%93intention_software_model (visited on 01/21/2020).

[47] Wikipedia Foundation Inc. *Classical Definition of Knowledge – Epistemology*. Aug. 26, 2014. URL: https://en.wikipedia.org/wiki/Epistemology#/media/File:Classical_definition_of_Kno.svg (visited on 01/27/2020).

[48] Wikipedia Foundation Inc. *Computer Security*. Feb. 17, 2020. URL: https://en.wikipedia.org/wiki/Computer_security (visited on 02/19/2020).

[49] Wikipedia Foundation Inc. *Exploit (computer security)*. Nov. 23, 2019. URL: https://en.wikipedia.org/wiki/Exploit_(computer_security) (visited on 01/21/2020).

[50] Wikipedia Foundation Inc. *Mankind*. Dec. 19, 2019. URL: https://en.wikipedia.org/wiki/Mankind (visited on 01/15/2020).

[51] Wikipedia Foundation Inc. *Nature*. Jan. 14, 2020. URL: https://en.wikipedia.org/wiki/Nature (visited on 01/15/2020).

[52] Tsau Young Lin, Qing Liu, and Y. Y. Yao. "Logics Systems for Approximate Reasoning: Approximation via Rough Sets and Topological Spaces". In: *ISMIS*. 1994.

[53] Aditya P Mathur and Nils Ole Tippenhauer. "SWaT: A water treatment testbed for research and training on ICS security". In: *International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*. IEEE. 2016, pp. 31–36.

[54] Ueli Maurer. "Information-theoretic cryptography". In: *Annual International Cryptology Conference*. Springer. 1999, pp. 47–65.

[55] J Ellis McTaggart. "The unreality of time". In: *Mind* (1908), pp. 457–474.

[56] Peter Mell, Karen Scarfone, and Sasha Romanosky. "A complete guide to the common vulnerability scoring system version 2.0". In: *Published by FIRST-forum of incident response and security teams*. Vol. 1. 2007, p. 23.

[57] Committee on National Security Systems (CNSS). "Glossary No 4009". In: *National Information Assurance (IA) Glossary* (Apr. 6, 2015). URL: https://rmf.org/wp-content/uploads/2017/10/CNSSI-4009.pdf.

[58] Jakob Nielsen. *Stop password masking*. June 22, 2009. URL: https://www.nngroup.com/articles/stop-password-masking/ (visited on 01/22/2020).

[59] NIST. *SP 800-160 Vol. 1 – Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems*. Mar. 21, 2018. URL: https://csrc.nist.gov/publications/detail/sp/800-160/vol-1/final.

[60] Plato et al. *Plato in Twelve Volumes: With an English Translation*. Ed. by Heinemann H. Harvard University Press, 1914. URL: http://www.perseus.tufts.edu/hopper/text?doc=Perseus%3Atext%3A1999.01.0172%3Atext%3DTheaet.%3Apage%3D201#note1.

[61] R. Karl Popper. *Conjectures and refutations: The growth of scientific knowledge*. New York London, 1962.

[62] Saikeerthi Rachavelpula. "The Category of Mereotopology and Its Ontological Consequences". In: *University of Chicago Mathematics Research Program*. Ed. by Michael Neaton and Peter Peter. 2017.

[63] MIT Research and Engineering (MITRE). *Common Vulnerabilities and Exposures (CVE)*. URL: `https://cve.mitre.org/` (visited on 01/27/2020).

[64] Patricia A Robinson. *Writing and designing manuals and warnings*. CRC Press, 2019.

[65] Marco Rocchetto, Martín Ochoa, and Mohammad Torabi Dashti. "Model-Based Detection of CSRF". In: *Proceedings of the ICT Systems Security and Privacy Protection – IFIP TC International Conference, SEC*. 2014, pp. 30–43.

[66] Marco Rocchetto and Nils Ole Tippenhauer. "CPDY: extending the Dolev-Yao attacker with physical-layer interactions". In: *International Conference on Formal Engineering Methods*. Springer. 2016, pp. 175–192.

[67] Marco Rocchetto, Luca Viganò, and Marco Volpe. "An interpolation-based method for the verification of security protocols". In: *Journal of Computer Security* 25.6 (2017), pp. 463–510.

[68] SAE. *Cybersecurity Guidebook for Cyber-Physical Vehicle Systems*. 2016. URL: `https://www.sae.org/standards/content/j3061_201601`.

[69] SAE. *DO -326A and ED-202A : An Introduction to the New and Mandatory Aviation Cyber-Security Essentials C1949*. 2019. URL: `https://www.sae.org/learn/content/c1949/`.

[70] Spyridon Samonas and David Coss. "THE CIA STRIKES BACK: REDEFINING CONFIDENTIALITY, INTEGRITY AND AVAILABILITY IN SECURITY." In: *Journal of Information System Security* 10.3 (2014).

[71] Katia Santacà et al. "A Topological Categorization of Agents for the Definition of Attack States in Multi-agent Systems". In: *Proceedings of the European Conference on Multi-Agent Systems and Agreement Technologies (EUMAS)*. 2016, pp. 261–276.

[72] Barry Smith. "Mereotopology: A theory of parts and boundaries". In: *Data & Knowledge Engineering* 20.3 (1996). Modeling Parts and Wholes, pp. 287–303.

[73] Eugene H Spafford. *Quatable Spaf*. Dec. 4, 2019. URL: `https://spaf.cerias.purdue.edu/quotes.html` (visited on 02/19/2020).

[74] Richard Stallman. *The Hacker Community and Ethics: An Interview with Richard M. Stallman*. 2002. URL: `https://www.gnu.org/philosophy/rms-hack.html` (visited on 01/22/2020).

[75] International Organization for Standardization (ISO). *ISO/IEC/IEEE 15288:2015, Systems and Software Engineering – System Life Cycle Processes.* May 2015. URL: https://www.iso.org/standard/63711.html (visited on 02/20/2020).

[76] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). "Information technology – Security techniques – Information security management systems – Overview and vocabulary". In: (2009). URL: http://standards.iso.org/ittf/PubliclyAvailableStandards/c041933_ISO_IEC_27000_2009.zip (visited on 01/22/2020).

[77] National Institute of Standards and Technologies (NIST). *National Vulnerability Database.* URL: https://nvd.nist.gov/ (visited on 01/22/2020).

[78] National Institute of Standards and Technologies (NIST). *Official Common Platform Enumeration (CPE) Dictionary Statistics.* URL: https://nvd.nist.gov/products/cpe/statistics (visited on 01/27/2020).

[79] Matthias Steup and Ram Neta. "Epistemology". In: *The Stanford Encyclopedia of Philosophy.* Ed. by Edward N. Zalta. Spring 2020. Metaphysics Research Lab, Stanford University, 2020.

[80] Keith Stouffer. *Guide to industrial control systems (ICS) security.* 82, pp. 16–16.

[81] *The Youtube Poll Expert.* 2019. URL: https://www.pollpro.com/ (visited on 02/17/2020).

[82] John Turri. "Is knowledge justified true belief?" In: *Synthese* 184.3 (2012), pp. 247–259.

[83] Mathieu Turuani. "The CL-Atse protocol analyser". In: *International Conference on Rewriting Techniques and Applications.* Springer. 2006, pp. 277–286.

[84] Achille C. Varzi. "On the Boundary Between Mereology and Topology". In: *Proceedings of the International Wittgenstein Symposium.* 1994, pp. 261–276.

[85] Ricardo Villadiego. *The Need For A Breakthrough In Cybersecurity.* Oct. 9, 2019. URL: https://www.forbes.com/sites/forbestechcouncil/2019/10/09/the-need-for-a-breakthrough-in-cybersecurity/#520e08839f1f (visited on 01/22/2020).

[86] David Von Oheimb and Sebastian Mödersheim. "ASLan++—a formal security specification language for distributed systems". In: *International Symposium on Formal Methods for Components and Objects.* Springer. 2010, pp. 1–22.