On Etiology of Cybersecurity

Marco Rocchetto
V-Research
Verona, Italy
Email: marco@v-research.it

Francesco Beltramini
V-Research
Verona, Italy
Email: francesco@v-research.it

***Abstract*—The objective of this research is the development 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. The core theories are the epistemological definition of knowledge, Beliefs, and Information, the Assertion-Belief-Fact framework, and mereotopology. 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) configuration. The configurations are reported as models satisfying the abstract system semantics. We use these results to predict all possible security weaknesses of a system, instead of relying on databases of known weaknesses, and to lay the foundations of a scientific theory of cybersecurity. We show a concrete applications of our theory to the risk assessment of an ad-hoc system.**

I. Introduction

In [22], 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 unfalsifiable". Given that, any theory which is not falsifiable by an empirical experiment is well known[1] 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 [21]. Herley, e.g. in [20], 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 [22], no solution is provided nor envisioned. On the contrary, the goal of this work is to lay the foundations of a scientific cybersecurity theory.

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 [22] 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 (against the security properties verified). 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 generality of the results, since they tackle a specific step (and not the first) of the engineering process (of systems). Therefore, the theories underlying the verification are based on assumptions which non-evidently apply to a general security theory. As an example, the so called Dolev-Yao attacker model[2] [12] only applies to specific instances (often called scenarios) and abstraction of the protocol. This, in turn, creates a false sense of security since requires non-justifiable assumptions on the abstraction of the system of which security is verified. More specifically, for the formal security verification of a system in which two or more agents are communicating, a formalized scenario needs to be defined by a modeler 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 [15]), (iii) honesty/dishonesty of the peers (e.g. in the ASLan++ language [60]), and (iv) the abstraction of the cryptographic primitives (e.g. ProVerif vs CryptoVerif [5]). While many tools and theories improves those issues, there is no agreement on which should be the definitive approach (if any). More importantly, some of the choices made by the modelers/engineers using those formal approaches may completely change the results of the formal verification of the system (an interesting example on how

[1] "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 [37]

[2] 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.

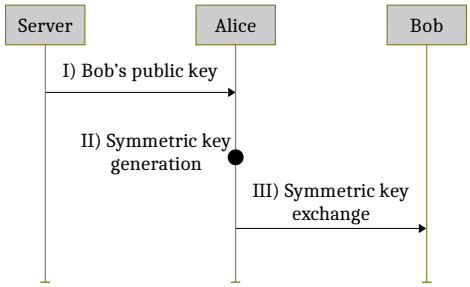

Figure 1. Abstraction of an ad-hoc esemplificative protocol execution

difficult it is to even just compare the approaches is given in [9]). For example, in Figure 1, under the perfect cryptography assumption[3] and assuming that no violation to any security property is done after message I), 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-falsifiable theory of system security

a) Structure: We start, in Section II, by detailing the problem statement, reporting a literature review on the main concepts and definitions related to security. We formulate a security hypothesis in Section III; which we use to propose a theory on system security in Section IV. In Section IV-C, we apply our theory for the tool-assisted security risk assessment of a CPS (with an ad-hoc example, based on the SWaT testbed [32]). This application shows how our theory can be used to predict all of the possible security weaknesses of a system, allowing the falsification of our theory. In fact, if any security weaknesses were to be found in a system and not predicted by our theory, the theory could be declared incomplete. Similarly, if a security weakness would be predicted by our theory but found to be impossible to realize, our theory could be declared as wrong.

II. Literature Review

In most of the natural languages 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[4] 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[5] (incidents are intentional while accidents are not) of the causes that generates a threat; namely, nature is believed not to have malicious intents (but unfortunate causes-effects) while threats generated by mankind are malicious[6]. This conclusion seems to be against a general formulation of a cybersecurity theory; how can we define a theory that predicts what a human being will do? Cybersecurity attacks seems to be related to the creativity of the attacker and then unpredictable. Currently, our understanding of cybersecurity is stored into a network of databases of weaknesses (e.g. CWE[10]), vulnerabilities (e.g. CVE[8], NVD[51]), and attacks (e.g. CAPEC[7], ATT&CK[39]). Systems are nowadays tested against known attacks, ignoring that considering a system secure if no known attack is possible is against the scientific method. As an example, if we have a system which is secure against all of the vulnerabilities in the CVE we just need to wait for a month to have 1000+ more vulnerabilities to test [52]. If even just one of those new vulnerabilities affects our system, that falsify the hypothesis that a system is secure if no known attacks apply. We argue that this "implement and test" approach should be changed. Even tough the field of security protocols is still lacking of a general cybersecurity theory, the secure-by-design approach they apply can be considered as going into the opposite direction of the "implement and test" approach.

A. Terminology

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 are defined (e.g. in standards such as the IEC 61508 or 61511[1]), are based on empirical tests and measurements (therefore should be considered hypothesis and not definitions). While reasoning by induction[7] based on the empirical observation should be avoided, since it may easily lead to false beliefs, this approach is often justified by the supposed impossibility of defining a theory that correctly predicts failures. A failure of a wire due

[3]As defined in [42]: "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 [57, 3, 2, 43]"

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

[5]"The belief–desire–intention software model (BDI) is a software model developed for programming intelligent agents."[26]. In the BDI model, the intents represents the deliberative state of an agent which determines the choice of that agent on what to do.

[6]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 even if due to the lack of skills.

[7]"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 establish by way of induction the universal proposition, the fall into the error of circular reasoning, since they are establishing 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 [13]

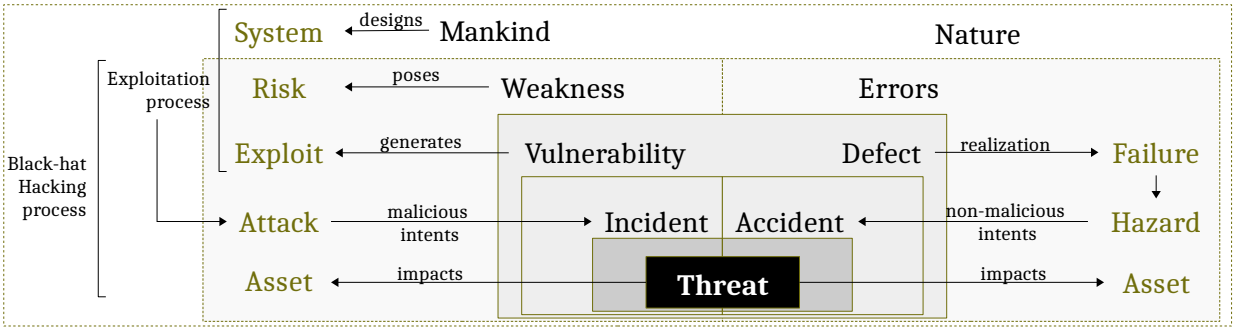

Figure 2. Overview of keywords related to security and safety

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 for security (e.g. for security testing). An overview on 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 security.

- *Mankind* "refers collectively to humans" [28], 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)[29].
  - There exist several terms to refer to an *attacker*, i.e. threat agent or threat source, considering those terms to be semantically equivalent. From now on, 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 materializes a threat.
- *Vulnerability*[8], as defined in [35] (and adopted in [4]), 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 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[35] may be identified with both Nature and Mankind, not differentiating between safety and security.

As depicted in 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 [...]"[35]. For example, a software or procedure that takes advantage of the vulnerability causing an *attack* to the system 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).

- *Weakness*. The definition given by the MITRE in [16] 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 a 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.

- *Causality* refers to the causality principle; defined in[11] 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". One can consider the causality principle to be formalized by the K Modal Logic.
- An *Exploit*[9] "[...] (from the English verb to exploit, meaning to 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

[8]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" [6]) highlighting how commonly this word is used without a proper supporting semantics.

[9]We note that the term exploit is only used as a verb in[50]

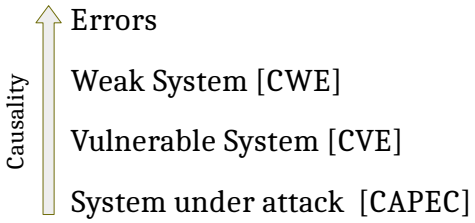

Figure 3. Etiology of cybersecurity

on computer software, hardware, or something electronic (usually computerized)."[27].

- 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 (as, e.g. in [48]).
- A *Threat*, as defined in[35], 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"[41] (i.e. a characteristic of an object which hinders its proper usability).
- *Failure*, as defined in[25] as "a state of inability to perform a normal function". The term is structured and detailed in [35, 14] but relying on an abstract notion of failure without a specific definition.
- *Hazard*, "a potential source of harm"[14].

Our literature review shows that most of the definitions relates insecurity to dis-honesty (also called maliciousness or adversarial) of an agent (often called adversary or attacker). This, however, just moves the problem of defining what cybersecurity is as the problem of defining what an dishonest agent can or cannot do in a system[10] As in the Dolev-Yao theory, we may correlate "being dishonest" to "not following the intended behavior/rules". In the case of a generic system, a dishonest agent is, therefore, any agent that doesn't follow the intended behavior (or functionality or logic) of the system. Given the generality of the definition, and its high level of abstraction, we may conclude that the hypothesis seems evident. For example, a software can be considered an agent of the system and whenever it has a bug, it can be exploited causing an Incident. However, this hypothesis has lead to the un-testable conclusion that the dishonest behaviors of agents cannot be defined in general (e.g. due to the heterogeneity of agents and systems) and that huge repository of dishonest behaviors should be kept as definition, such as CAPEC[7]; or that dishonesty of an agent with respect to a security protocol should be defined as a number of predefined actions as in [57, 3, 2, 43] (to name a few). As depicted in Figure 3, in order to define a theory on cybersecurity and on the malicious behaviors of agents we noticed that Attack Patterns (or Attacks) are caused-by the presence of Vulnerabilities in the system. Those Vulnerabilities are, in turn, cause-by the presence of Weaknesses in the system. Weaknesses are Errors in the design or implementation of a system. Therefore, a theory on cybersecurity should first predict the Errors in a system design.

[10]where an agent is any virtual or physical entity of the system or using the system (e.g. a device, a software, or a human being) and dishonesty is not necessarily related to malicious motivation but also to incompetence or lack of skills.

III. Security Hypothesis – ABF Theory of Agents

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 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 development 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 because the *intended (or nominal) behavior* of the system is not precisely defined in the specification at the very early stages of the engineering process.

As a reference, we consider the following steps of the engineering process and our objective is to be able to predict the security requirements instead of axiomatically assume them.

1) *System Specification*, where the *functional and physical requirements* are defined
2) *Architecture Design*, where the specification is structured into *functional and physical architectures*
3) *Security Risk Assessment*, where the potential *weaknesses* (errors) are identified and, consequently, *security requirements* are captured.

Given that, in our investigation, we changed the focus from the attacker to the potential designs of a system, we start by defining a general framework for the definition of a system. On top of this framework we will identify system weaknesses as potential design errors.

A. Systems as Multi-Agent Systems

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."[49]. 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.

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[11].

- In [23], 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 [24], Hintikka describes the concept of Information and the difference with Knowledge and Belief.
- In [46], the authors defines an agent as a tuple of Assertions, Beliefs, and Facts.

For our argument, as in [46], an agent is composed by its knowledge, beliefs, and the information or assertions it provides; where knowledge is defined, as in [53], 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*, and the only objective of Information is to exchange beliefs between agents. As defined in [24], "Information is specified by specifying which alternatives concerning the reality it admits and which alternatives excludes". This means that if we consider 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. 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. The difference between Knowledge and Information as used by Hintikka, and Facts and Assertions as used in [46], are not precisely discussed in [46]. Furthermore, those epistemological concepts are difficult to define [56][12]. For example, Hintikka in [24] states that "A purely logical definition of information is impossible". However, a detailed formalization of those concepts is out of the scope of this paper.

We now define them abstractly, inspired by the ABF-theory defined in [46] and, in Section IV we detail them as engineering concepts.

[11]It is interesting to notice that in [13], 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.". The author than proceeds showing that even in this framework, knowledge still is non-apprehensible.

[12]"*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 [36]

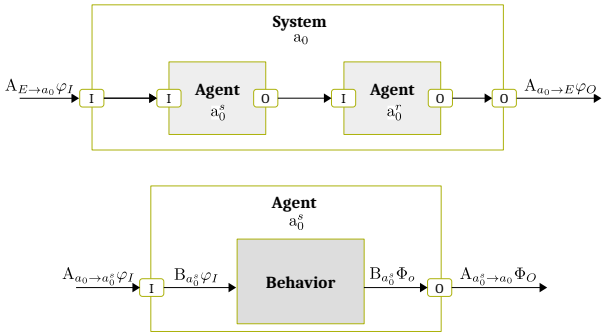

Figure 4. Example of system and agent structure

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 [13]), and
3) We consider a set of axiomatic *Facts* (definitory rules[14]) instead of considering the more general epistemic definition of Knowledge. Specifically, Facts describes:
   - The functional architecture of each agent
   - The physical/structural (HW/SW) architecture of each subsystem of agents and agent within a subsystem

We give a graphical representation of (sub-)system and agent in Figure 4

B. Mereo-topological Reasoning

Similarly to[46], we define agents as a meronomy (an hierarchy of Part-Whole relations) over the constituent that we previously defined (assertions, beliefs, and facts), based on a standard definition of mereology, i.e. based on the definition of Parthood relation between *Parts*. Due to the necessity of considering different types of Part (as we'll show afterwards) we extend the mereology to a mereo-topology[47, 59, 38], considering the relations in Table I. For the sake of readability, we use the term *Region* both to refer to a mereological Part and to a topological Region. Our aim is to create a meronomy instead of the taxonomies such as the one provided in[51, 40] with the so that we don't need to rely on a scoring system such as the CVSS[34]. Instead, we want to precisely calculate the number of insecurity configurations as a scoring system.

[13]"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 [24]

[14]Facts are definitory rule as they don't define how a real system is finally implemented but how it should be, trough a series of requirements. Those requirements may not hold, through insecurity, in the implemented system.

RCC3	RCC5	RCC8	Terminology	Notation	Definition
			Connects with	$C(X, Y)$	$X \subseteq Y$
			Disconnected from	$\neg C(X, Y)$	$X \not\subseteq Y$
			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**	$NTPPi(X, Y)$	$NTPP(Y, X)$

Table I
RCC3, RCC5, AND RCC8 RELATIONS BETWEEN REGIONS X, Y AND Z

A mereotopology, as defined e.g. in[38], is an ordered mathematical structure where the basic relation between Regions is the reflexive and symmetric *Parthood* relation $\subseteq$.

Definition 1. Parthood – *Given any pair of mereotopological Regions X and Y,*

1) Reflexivity: $\forall X.\ (X \subseteq X)$

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

The Parthood relation orders a universe of agents Ag by defining the so called *Connects with* (see in Table I) relation between Regions. We use the Region Connection Calculus (RCC), as defined in [30, 18], to provide an axiomatization of the mereo-topological concepts. 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. In Table I, we summarize the axioms of the Region Connection Calculus (see, e.g., [19]). We can now define a system over the mereotopology using the RCC calculus, as follows.

Definition 2. System State – *A CPS system (or a sub-system) state is defined as the tuple $s = \langle rcc(\mathcal{F}, \mathcal{B}), rcc(\mathcal{F}, \mathcal{A}), rcc(\mathcal{B}, \mathcal{A})\rangle$, where $\mathcal{A}, \mathcal{B}, \mathcal{F}$, are regions of assertions, beliefs (i.e. the beliefs generated by the behavior), and facts expressed as requirements respectively.*

As already presented in [46] it follows that, defining a 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 [46].

a) 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 regions, 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 regions, $\binom{n}{k}$. In our case, $\binom{n}{k} = 3$ since we consider 3 regions $(\mathcal{A}, \mathcal{B}, \mathcal{F})$, and all the relations considered in the RCC are binary. Hence, using RCC5 (with five different spatial relations) over three regions, we can theoretically define up to 125 different type of agents. However, only 54 of the 125 (as showed in [18]) 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 regions 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 5, 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.

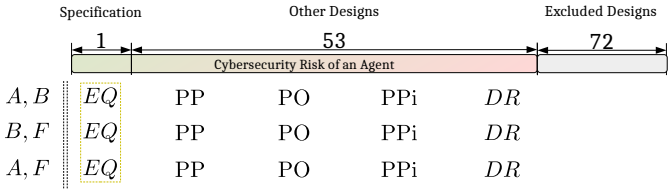

Figure 5. Cybersecurity risk for a single agent

C. 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

	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($\mathcal{B}$,$\mathcal{F}$)	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 II
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($\mathcal{A}$,$\mathcal{F}$)}

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 II, 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 security risk related to a generic agent.

As depicted in Figure 6, 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 one should consider different "degrees" of non-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 non-soundness. A similar argument can be done for completeness.

a) 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, as depicted in Figure 4, 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[15]:

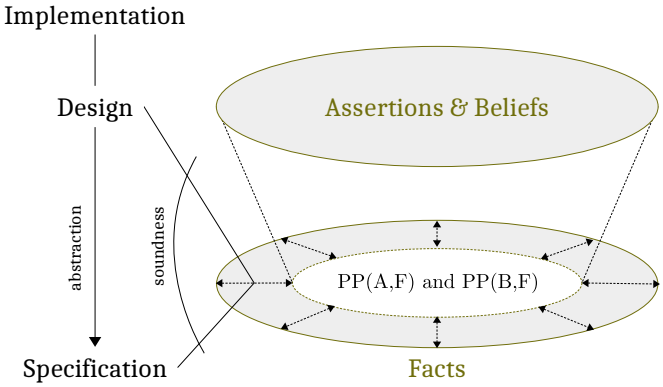

Figure 6. Example relation between Facts, and Assertions and Beliefs

- 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 complete.
- 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.

[15]"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[33]

b) rcc($\mathcal{A}$,$\mathcal{F}$) and rcc($\mathcal{B}$,$\mathcal{F}$) – Honesty and Competence: The relation of Assertions and Behavior with Facts determines the quality with respect to the nominal (specified) system. Given that Facts defines what needs to be true in the system, the relation of Assertions an Facts determines the degree of quality between the real information circulating in a system (or within an agent) and the one specified. 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 sharing truths[16]. The relation between Beliefs and Facts determines the competence (on the subjects 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).

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 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 [10]) is defined as $EQ(\mathcal{A}_a, \mathcal{B}_a)$ for an agent a while can be detailed as follows when multiple agents are considered.

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[10]) 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[10]) is defined as $EQ(\mathcal{B}, \mathcal{F})$.

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

Definition 3. 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 refers both to positive and negative consequences of uncertainty. Accordingly, we consider risk as follows.

Definition 4. 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 5, defined as follows.

Definition 5. 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).*

The risk matrix is often defined as a function between likelihood and impact of attacks. We suggest that security weaknesses are all equally likely to be exploited if there's a connection between the weakness and the asset that an attacker wants to impact. Therefore, a risk matrix should capture the number of insecure configurations of a system, rather than predicating over likelihoods of weaknesses exploitation.

IV. Prediction of Security Weaknesses

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 [45].
2) NIST 800-82 [54] – "guide to Industrial Control System (ICS) Security"
3) J3061:2016-1 [44] – "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"

However, standards do not describe in detail how to perform a security risk assessment and only vaguely define the overall objective. The overall objective of this first step of the cybersecurity engineering process is to provide an understanding of the potential security risks. Roughly speaking, a security risk assessment (e.g. CORAS[31]) methodology starts from the (i) identification of assets, (ii) determines the threat scenarios (correlating vulnerabilities, threats and incidents) e.g. threat

[16]"[...] 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[13]

diagrams in the CORAS approach, (iii) calculate the risk as a function between impact and likelihood of threat scenarios, and (iv) finally provides treatments (mandating a partial re-design) and residual risks. All the methodologies and tools we reviewed (e.g. Threatmodeler [55], CORAS [31], SECRAM [17]) relies on the expertise of the person who performs the risk assessment for the identification of threats and for the quantitative estimation of risks. As Herley puts it [21], "we don't have a test for security and we don't have a metric for security" while standards requires a security risk assessment which, in turn, mandates a metric to evaluate the security risk.

So far, we have reasoned on systems in the abstract, considering generic MAS; mapping Epistemological concepts to MAS. We now map MAS to CPS and show how this mapping allows us to predict security weaknesses in a system.

A. From Multi-Agent to Cyber-Physical Systems

As depicted in Figure 7, we relate MAS and CPS as follows.

- We consider a System as an hierarchy of agents. So, we map agents to *systems, sub-systems, or devices*, depending on the granularity of the design. For example, a modeler can model a specific device, as in Figure 7. In this case, the system is not decomposed into sub-systems or devices and the device is considered as an agent.
- Agents reason over Beliefs (i.e. transform Beliefs into other Beliefs) and for each *component of a CPS* (system, sub-system, or device) is composed by a *functional architecture* that transforms input-Beliefs into output-Beliefs.
- Components of a CPS have *ports* to exchange information with the outer environment (which may be a sub-system), similarly to Agents. The transfer of information in a CPS is defined by *channels*.
- The concept of Knowledge is related to the *requirements* that, in a factorial way, describe how the CPS shall behave and its *physical architecture*.

a) Input and Output Ports: Since the $\mathcal{A}, \mathcal{B}, \mathcal{F}$-theory is a theory of agents, we could consider ports as agents that allows the exchange of information between a channel and another agent. However, we considered a port as a special type of agent that 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 6. Input or Output Port – *A port forwards information from the outside of an agent's boundary to the inside (input-port) or vice versa (output-port). There exist two types of ports with the following behavior: an input-port changes transforms Assertions from a sender s ($\mathcal{A}_{s\to a}$) to Beliefs of an agent a ($\mathcal{B}_a$), while an output-port from belief to assertion.*

The *quality of a port* is determined by the rcc relation between the assertions received or sent and the belief, i.e. $rcc(\mathcal{A}_s, \mathcal{B}_a)$ for the input-port or $rcc(\mathcal{A}_r, \mathcal{B}_a)$. A port is, in fact, syntactic sugar to express the relation between Assertions and Beliefs. We only define secure input port but the definition of a secure output port is symmetrical.

Definition 7. Secure Input Port – *A secure input port ($s\boxed{\rightarrow}a$) always allow information as incoming assertions to flow from a sender s to the behavior of the recipient a (agent).*

b) Port Weaknesses: From our definition of ports, 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) *replace 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) *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)
W3) *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
W4) *injection port* ($s\boxed{\bullet\!\!\rightarrow}a$), where information coming from s is substituted with new information which becomes believed by a
W5) *selective port*, where some information passes the port and part is either:
W4.1) drop ($s\boxed{\circ\!\!\mapsto}a$),
W4.2) drop+insert ($s\boxed{\bullet\!\!\mapsto}a$).

Proof. An input port is, in the $\mathcal{A}, \mathcal{B}, \mathcal{F}$-theory, defined secure as long as the relation between the two regions of input assertions $\mathcal{A}$ and output beliefs $\mathcal{B}$ are equal, i.e. $EQ(\mathcal{A}, \mathcal{B})$. 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. $DR(\mathcal{A}, \mathcal{B})$. 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. We distinguish between Regions that contain Information and Regions that don't by writing the latter as $\emptyset$.

1) if $EQ(\mathcal{A}, \mathcal{B}))$ then either $\mathcal{A} = \mathcal{B} = \emptyset$ (no communication) or $\mathcal{A} = \mathcal{B} \neq \emptyset$ (forward communication)
2) if $DR(\mathcal{A}, \mathcal{B}))$ then $\mathcal{A} = \emptyset \oplus \mathcal{B} = \emptyset$ (we call *insert* the former and *full drop* the latter case), or $\mathcal{A} \neq \emptyset \wedge \mathcal{B} \neq \emptyset \wedge \mathcal{A} \neq \mathcal{B}$ which we call *replace* (i.e. drop and insert)
3) if $PP(\mathcal{A}, \mathcal{B}))$ then $\mathcal{B}$ contains and extend $\mathcal{A}$ which we call *injection*
4) if $PPi(\mathcal{A}, \mathcal{B}))$ then $\mathcal{A}$ contains and extend $\mathcal{B}$ which we call *selective drop*
5) if $PO(\mathcal{A}, \mathcal{B}))$ then a part of the $\mathcal{A}$ is contained in the $\mathcal{B}$ which is a combination of *selective drop and generation*

□

c) Communication Channels: In this work, we only consider mono-directional channels and communication but the extension to bi-directional channel can be considered as

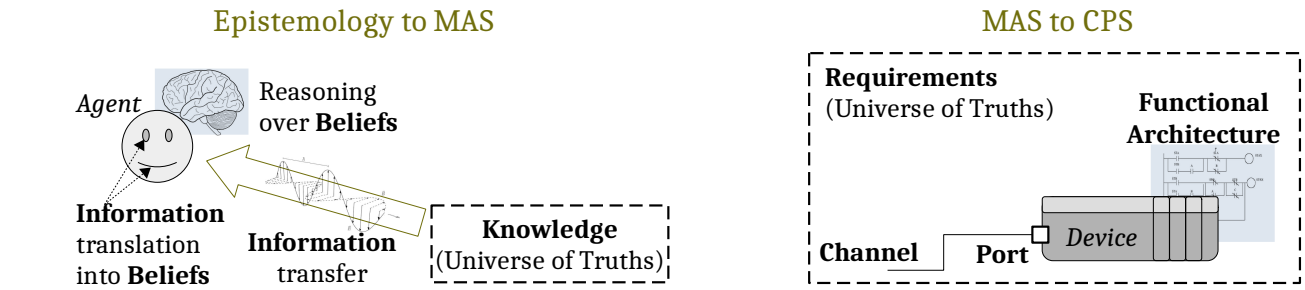

Figure 7. asd

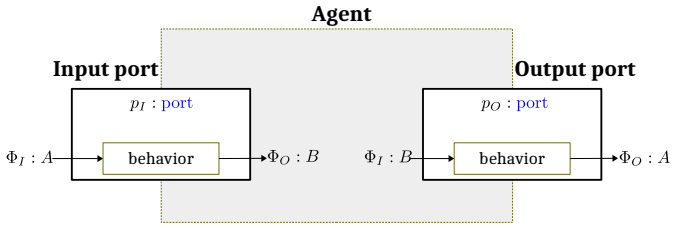

Figure 8. Input and Output Port of an agent

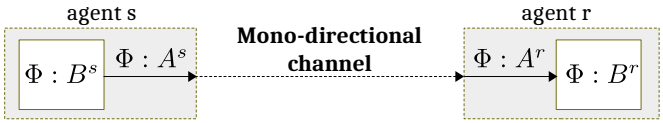

Figure 9. Communication over a Mono-directional Channel

the union of two unidirectional channels. A mono-directional channel is defined by the assertions sent or received (over the channel).

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 $\mathcal{A}, \mathcal{B}, \mathcal{F}$-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, we consider channels to be bottom-channels, defined as agents with the pre-defined behavior (i.e. defined in a axiomatic way) of forwarding any input-assertion as output-assertion, without modifying it[17].

Definition 8. 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. $rcc(\mathcal{A}_s, \mathcal{A}_r)$.

d) 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 output side; and we exclude the configuration with 2 secure types as input and output, -2); where only 43 are possible. For the sake of readability, we report 6 examples in the following but the proof by exhaustion over all the possible cases is straightforward.

W6) *secure output port* and *input drop port* ($s \rightarrow\!\circ\ r$),
W7) *secure output port* and *input insertion port* ($s \rightarrow\!\bullet\ r$),
W8) *output drop port* and *input drop port* ($s \circ\!\rightarrow\!\circ\ r$)
W9) *output drop port* and *input insertion port* ($s \circ\!\rightarrow\!\bullet\ r$)
W10) *output drop port* and *input secure port* ($s \circ\!\rightarrow\ r$)
W11) *output injection port* and *input secure port* ($s \bullet\!\rightarrow\ r$)

e) Security Weaknesses: It is important to note that all the results of the application of the $\mathcal{A}, \mathcal{B}, \mathcal{F}$-theory to channels (the analysis of the relation $rcc(\mathcal{A}_o, \mathcal{A}_i)$) lead to the same results of the analysis of a pair of an input and output port (i.e. $rcc(\mathcal{B}_1, \mathcal{A}_2)$ and $rcc(\mathcal{A}_2, \mathcal{B}_3)$). The same result can be obtained by analyzing the relation between the outputs of a functional block and the inputs of another functional block, where functional blocks are constituents of the functional architecture as described afterwards. As depicted in Figure 10, so far, we have considered Information generated by a Port P_I and then sent through a channel C to another (recipient) port P_O. In this scenario where Ports and then Channels are atomic (otherwise raising infinite regress), we can only consider the relations between Ports and Channel; considering both input-port to channel and channel to output-port. In fact, the Weaknesses of a channel are defined in terms of the Weaknesses of Ports. We now introduce the concept of functional architecture and its decomposition into functional blocks. Io order to define a functional block without encountering a infinitely recursive definition, we must reach the same conclusions as for the

[17]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.

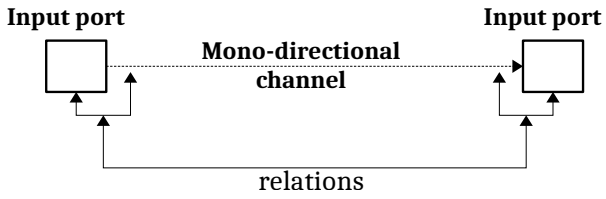

Figure 10. Relations between Ports and Channel

Channel. Therefore, describing the information as flowing over a channel or in a functional block is purely syntactic sugar.

We can summarize these results by saying that the relations between Assertions and Beliefs, Output Assertions of an Agent and Input Assertions of another Agent, or Output Beliefs of a Functional Block and Input Beliefs of another Block can *only* be affected by the following weaknesses: replace, drop, injection, insertion, selective drop, and selective drop + insertion. We can, then, deduce the following security properties to mitigate security weaknesses of a port or a channel (between ports, functional blocks, or both).

- *Order-preserving* – it shall be known if Information is *replaced*
- *Availability* – it shall be known if Information is *dropped* or *selectively dropped*
- *Integrity* – it shall be known if Information is *injected*
- *Authentication* – it shall be known if Information is *inserted*

f) Functional Architecture: A Functional Architecture 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 executing an abstract undefined behavior, of which we only observe the inputs and the resulting outputs (Beliefs).

Definition 9. Functional Block and Architecture – *A functional block of an agent takes a region of input-Beliefs and outputs a region of output-Beliefs. A functional architecture is an interconnected system of functional blocks.*

It is evident that the quality of a functional block cannot be determined by the difference between its inputs and outputs (as we did for ports and channels). The behavior of a functional block cannot be determined in general; since any functional block will have its own purpose based on functional requirements. Therefore, while a port semantics is determined by the relation between Assertions and Beliefs, the semantics of a Functional Block is determined by the relation between Facts/Requirements and I/O Beliefs.

In other words, a functional block is a generic agent with no pre-defined general behavior (while ports and channels have a pre-defined behavior).

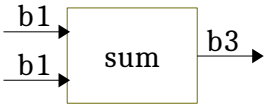

Figure 11. Example of a Functional Block

W50) $PO(\mathcal{B},\mathcal{F})$ the component has a Byzantine behavior where occasionally outputs the expected output given the correct inputs. Not all the inputs are handled properly, nor all the expected outputs are always generated when correct inputs are given.

W51) $PP(\mathcal{B},\mathcal{F})$ part of the expected outputs are not generated in response to the correct inputs

W52) $PPi(\mathcal{B},\mathcal{F})$ the components correctly performs the expected behavior when the correct inputs are provided but is subject to input injections

W53) $DR(\mathcal{B},\mathcal{F})$ the component never performs the expected behavior (e.g. physical damage)

g) Requirements as Facts: During the specification phase, for any agent, channel, port, functional block and architecture, there may exist a factorial requirement (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 IV and depicted in Figure 6, Facts are strategic rules that defines how the system shall behave (by specification), while reality may be shown to be insecure.

As an example, consider a functional block that performs the summation of two inputs, as in Figure 11, defined by the requirement $r := b_3 = b_1 + b_2$. The possible relations between the behavior of the functional block and the requirements (i.e. $rcc(\mathcal{B},\mathcal{F})$ is determined by the relations between the I/O Beliefs $sum(b_1, b_2) = b_3$ and the requirement $sum(b_1, b_2) = b_1 + b_2$, as follows.

- $EQ(\mathcal{B},\mathcal{F}) = EQ(\mathcal{B}_3, \mathcal{B}_1 + \mathcal{B}_2)$, where $\mathcal{B}_3$ represents the Region of all the outputs of the functional block sum while the Region $\mathcal{B}_1 + \mathcal{B}_2$ represents the expected outputs of an ideal implementation of the requirement r. Therefore, the functional block correctly implements the requirements.
- $DR(\mathcal{B},\mathcal{F}) = DR(\mathcal{B}_3, \mathcal{B}_1 + \mathcal{B}_2)$, the function block never implements the requirements.
- $PP(\mathcal{B},\mathcal{F}) = PP(\mathcal{B}_3, \mathcal{B}_1 + \mathcal{B}_2)$, there exist some inputs for which the output is incorrect.
- $PPi(\mathcal{B},\mathcal{F}) = PPi(\mathcal{B}_3, \mathcal{B}_1 + \mathcal{B}_2)$, there exists some outputs that are not the result of the summation of the inputs but given the correct inputs the function always outputs correctly.
- $PO(\mathcal{B},\mathcal{F}) = PO(\mathcal{B}_3, \mathcal{B}_1 + \mathcal{B}_2)$, the component has a Byzantine behavior where occasionally outputs the expected output given the correct inputs. Not all the inputs are handled properly, nor all the expected outputs are always generated when correct inputs are given.

B. Security and Insecurity of a System

Hypothesis 1. System Security Design – *A system security design is given by a precise system specification over the physical and functional architectures, that uniquely, i.e. tested against the range of design possible in the $\mathcal{ABF}$-theory, defines the design built on top of those requirements.*

Hypothesis 2. System Insecurity Design – *If, given a system specification as a collection of requirements on the system, there exist a non-unique design with respect to those requirements, the number of equivalent designs that fulfills the requirements quantitatively defines the magnitude of insecurity of a system design.*

Based on these hypothesis, security (S) can be expressed as the following equation.

$$S = 1 - (\prod_{a \in Ag} r^{\binom{n}{k}} - \delta_T)$$

where $r^{\binom{n}{k}}$ represents the number of possible configurations (permutations) of an agent based on the number r of relations (e.g. EQ, DR, PO) with arity k between n different regions in the $\mathcal{ABF}$ theory. δ_T represents the number of configurations which are not satisfiable with respect to the logical theory defining the algebraic structure (topology) and constraints of the calculus (RCC). Ag is the set of all the agents in the system under evaluation. The constant 1 represents the nominal configuration of the system and then the only secure configuration. The product represents all the unintended, insecure, configurations.

C. Security Risk Assessment

In order to test our theory we implemented a tool-chain (open-source under the AGPLv3 license and available at [58]) for the identification of weaknesses and the precise calculation of potential insecure configurations. The engineering of the $\mathcal{ABF}$ is summarized in the UML Class diagram in Figure 16 (in appendix).

As depicted in Figure 12, the security risk assessment process starts with the definition of the use cases and architectural requirements; the specification of a system. In our process, the specification is manually translated into a UML design where:

- a *Deployment Diagram* describes the *Physical Architecture*. Each agent is defined as a Node with (physical) Ports, and Agent's ports are connected via Information Flow connectors, representing the physical channel.
- a *Functional Architecture* is linked to each agent in the deployment diagram and is defined by an *Object Diagram*. The Object Diagram is composed by Instances of Functional Blocks, connected via Information Flow connectors.
- the connection between the two diagrams is implemented by "sockets", functional blocks connected to a physical port.

The tool generates a graph-like internal structure which represents the specification (called ABF graph). The ABF

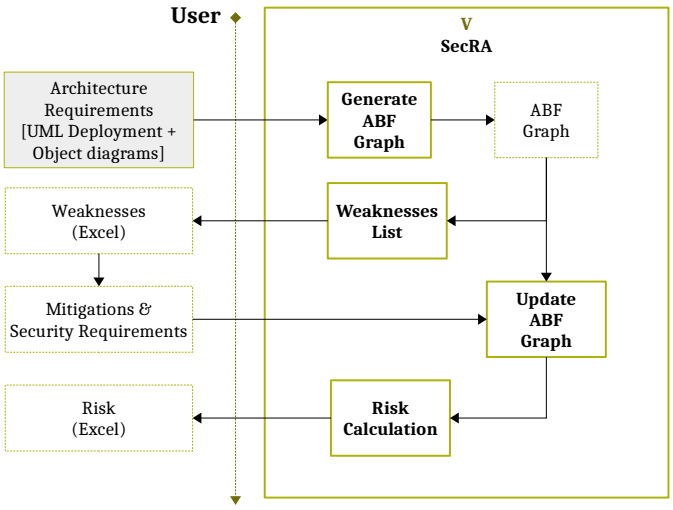

Figure 12. Security Risk Assessment Tool

graph defines the system as a number of regions of assertions, beliefs, and facts. Those regions are connected by a generic relation which is evaluated as follows. The graph is translated into a logical formula that represents the specification in the $\mathcal{ABF}$-theory and, along with the axiomatization of the RCC5 calculus, is given as input to the Z3 SMT solver. The solver identifies all possible configurations of the system and, in turn, to identify all potential weaknesses. The internal structures of the tool can be viewed as PDF and the results are reported into an Excel file. The Excel file also report the total number of configurations as indicating the security risk associated to the specification. A user can change the status of each weaknesses in the Excel file, from the initial (default) status (open) to "mitigated". As a consequence, the risk is re-calculated on-the-fly, i.e. without the need of running the tool again, based on annotations and formulas in the Excel file.

In our approach, security requirements are not imposed by the specification but are automatically extracted by our tool as potential weaknesses. Those weaknesses are related to the different insecure configurations of the specified system.

The risk matrix is often defined as a function between likelihood and impact of attacks. We suggest that security weaknesses are all equally likely to be exploited if there's a connection between the weakness and the asset that an attacker wants to impact.

a) Case Studies: We report the results of the evaluation of a water level reader (sensor ad-hoc example). As in Figure 13, we defined 2 agents: sensorInTank and sensorBoard, which represent the physical reader that needs to be placed in a tank of water, and the board that interprets the readings and outputs them as digital signals. The two components are connected by a wire. In Figure 14 we report the functional architecture that receives the incoming communications from the sensor in the tank, and communicates them encrypted. The results, summarized in Figure 15 reports more than 16 millions possible scenarios in which at least one component diverge

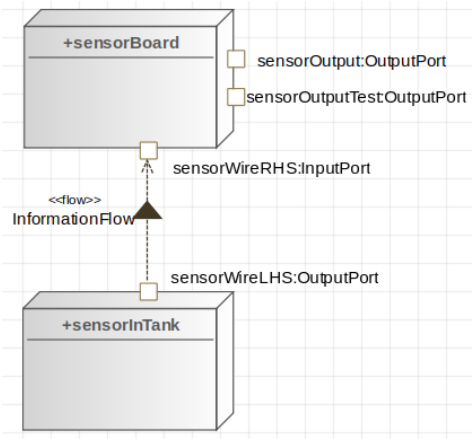

Figure 13. Water Level Reader Deployment Diagram

from the specification.

V. Discussions

a) Abstraction Level: The level of abstraction is always a concern when it comes to the definition of a system model. In this paper, we justified our assumptions on the atomicity of ports, channels, and functional blocks but nothing prevents a modeler to consider other atomic element of the system specification and design. Those additional elements may detail the system and may change the syntactic sugar we defined; however, it won't change the underlying (semantic) reasoning unless another calculus is used (instead of RCC). Furthermore, other Infosec properties may be considered, such as authenticity or non-repudiation. We, however, leave them as future development.

b) $\mathcal{ABF}$ Falsification: The theory we developed can be falsified by providing an experiment that shows a weaknesses not predicted by the $\mathcal{ABF}$-theory, or by showing that the weaknesses predicted for a certain experiment are impossible to realize. This theory, however, needs to be detailed to be applicable to a full engineering process (we only cover the first steps). We propose the following complete process as a draft for potential next steps in this direction.

1) System Specification
2) Architecture Design
3) Security Risk Assessment and Security Requirement Extraction
4) Design Refinement, where the semantics of the functional behavior and communication protocols are defined
5) Security Requirement Extension, based on the design refinement
6) Verification of Security Requirements (data flow, behaviors, & protocol logic)
7) Software Prototype Implementation/Synthesis
8) Automated Security Testing (abstract test-suite generation & concretization)
9) Hardware Prototype Implementation
10) Automated Security Testing
11) Penetration Testing

c) Relation with DY: The DY attacker model has been extensively used for the identification of logical weaknesses in security protocols. Our theory should predict all possible weaknesses which, in turn, should enable all the attacks identified by the DY model. A proof that the two theories/models correctly predicts (or lacks in predicting) weaknesses and related protocol flaws would connect risk assessment and protocol verification, the first two steps of the secure engineering of systems.

VI. Conclusion and Future Works

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 conclude that the problem of security is a problem related to the many possible design and, in turn, implementation given a specification. The problem is correlated 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.

Acknowledgment

We thank Katia Santacà for the help in the development of the initial ideas, and for the helpful discussions that allowed us to make this step.

Appendix

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] 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.

[3] 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.

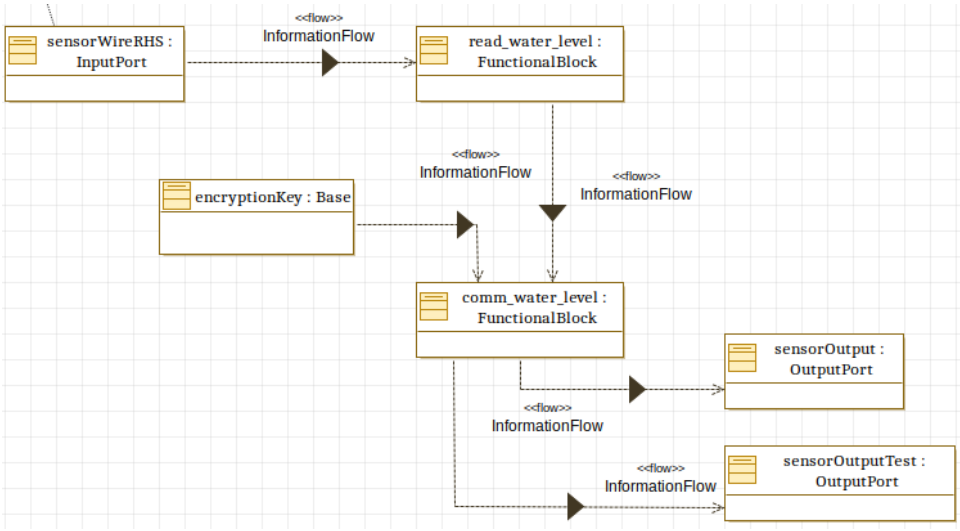

Figure 14. Sensor Board Object Diagram

RISK				16777216
The total risk is the total number of configurations of the system				
B	C	D	E	F
Agent	**Component**	**Comp. Type**	**Weakness**	**Status**
sensorBoard	sensorWireRHS	inputport	selectively drops inputs and inserts new malicious data	open
root	sensorWireLHS2sensorWireRHS	channel	selectively drops inputs and inserts new malicious data	open
sensorInTank	sensorWireLHS	outputport	selectively drops inputs and inserts new malicious data	open
sensorInTank	sensorWireLHS	outputsocket	the component has a Byzantine behavior where occasionally outputs the expected output given the correct inputs. Not all the inputs are handled properly, nor all the expected outputs are always generated when correct inputs are given.	open

Figure 15. Partial View of the results in the Excel file

[4] 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.

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

[6] 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.

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

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

[9] Cas JF Cremers, Pascal Lafourcade, and Philippe Nadeau. "Comparing state spaces in automatic security protocol analysis". In: *Formal to Practical Security*. Springer, 2009, pp. 70–94.

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

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

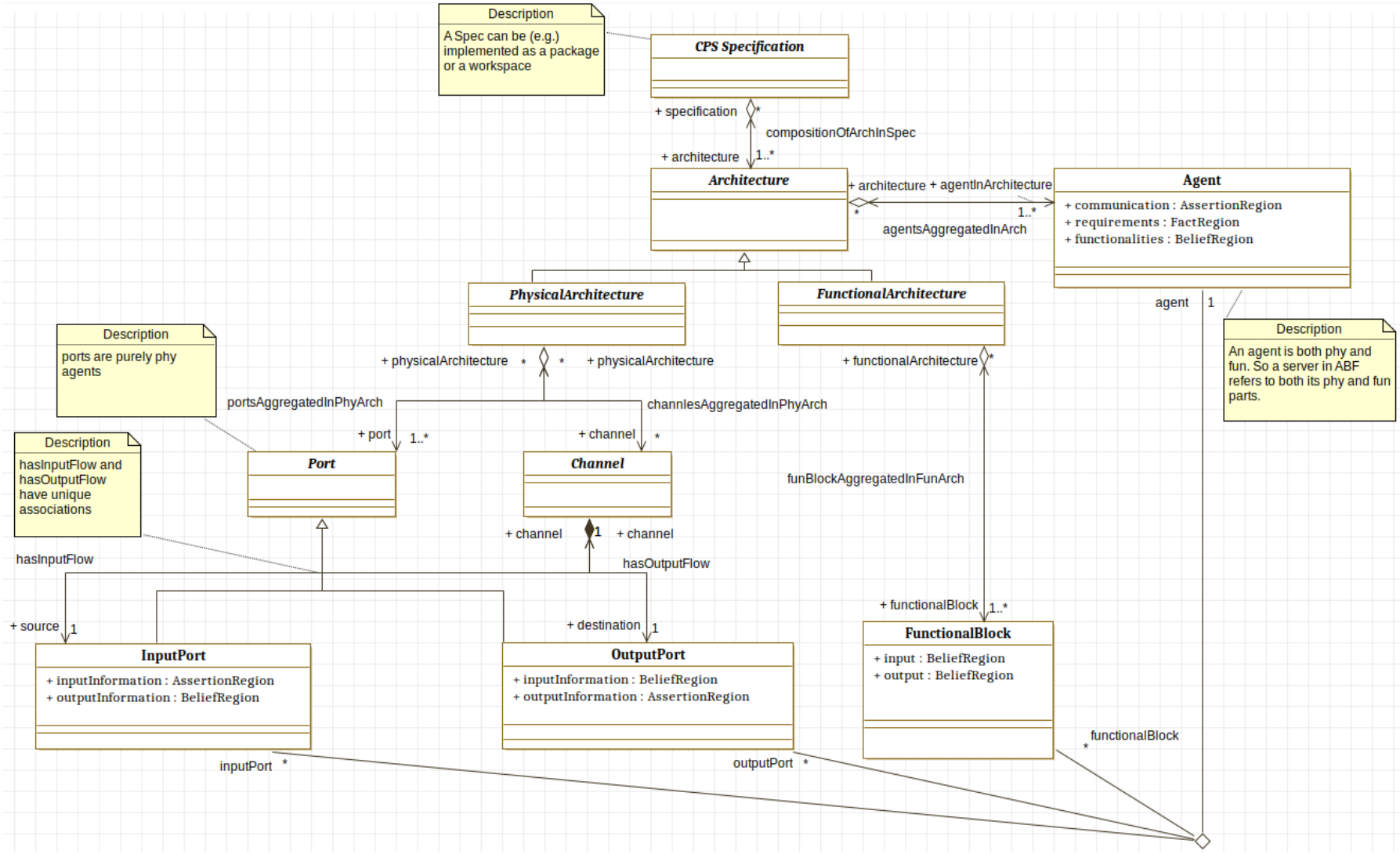

Figure 16. Security Risk Assessment – Class Diagram

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

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

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

[15] 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.

[16] *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).

[17] Martina de Gramatica et al. "The role of catalogues of threats and security controls in security risk assessment: an empirical study with ATM professionals". In: *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer. 2015, pp. 98–114.

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

[19] 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.

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

[21] 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).

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

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

[24] 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.

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

[26] 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).

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

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

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

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

[31] Mass Soldal Lund, Bjørnar Solhaug, and Ketil Stølen. *Model-driven risk analysis: the CORAS approach*. Springer Science & Business Media, 2010.

[32] 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.

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

[34] 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.

[35] 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.

[36] 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.

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

[38] 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.

[39] MIT Research and Engineering (MITRE). *ATT&CK*. URL: https://attack.mitre.org/ (visited on 06/09/2020).

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

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

[42] 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.

[43] 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.

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

[45] 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/.

[46] 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.

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

[48] 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).

[49] 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).

[50] 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).

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

[52] National Institute of Standards and Technologies (NIST). *NVD Dashboard*. URL: https://nvd.nist.gov/general/nvd-dashboard (visited on 06/09/2020).

[53] 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.

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

[55] Threatmodeler. *ThreatModeler*. URL: https://threatmodeler.com/ (visited on 06/09/2020).

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

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

[58] V-Research. *V-Research Cybersecurity Repository*. URL: https://github.com/v-research/cybersecurity (visited on 06/09/2020).

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

[60] 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.