Attack-Surface metrics, OSSTMM and Common Criteria based approach to "Composable Security" in Complex Systems

ANDREA FIASCHETTI, ANDREA LANNA, MARTINA PANFILI, SILVANO MIGNANTI, ANTONIO PIETRABISSA, FRANCESCO DELLI PRISCOLI
Department of Computer, Control and Management Engineering (DIAG)
"Sapienza" University of Rome
Via Ariosto 25, 00185 Rome
ITALY
surname@diag.uniroma1.it http://www.dis.uniroma1.it

ROBERTO CUSANI, GAETANO SCARANO
Department of Information, Communication and Electronic Engineering (DIET)
"Sapienza" University of Rome
Via Eudossiana 184, 00184 Rome
ITALY
name.surname@diag.uniroma1.it http://www.diet.uniroma1.it

ANDREA MORGAGNI
Selex Electronic Systems (a Finmeccanica Company)
Via Laurentina 760, 00143 Rome
ITALY
andrea.morgagni@selex-es.com http://www.selex-es.com

Abstract: - In recent studies on Complex Systems and Systems-of-Systems theory, a huge effort has been put to cope with behavioral problems, i.e. the possibility of controlling a desired overall or end-to-end behavior by acting on the individual elements that constitute the system itself. This problem is particularly important in the "SMART" environments, where the huge number of devices, their significant computational capabilities as well as their tight interconnection produce a complex architecture for which it is difficult to predict (and control) a desired behavior; furthermore, if the scenario is allowed to dynamically evolve through the modification of both topology and subsystems composition, then the control problem becomes a real challenge. In this perspective, the purpose of this paper is to cope with a specific class of control problems in complex systems, the "composability of security functionalities", recently introduced by the European Funded research through the pSHIELD and nSHIELD projects (ARTEMIS-JU programme). In a nutshell, the objective of this research is to define a control framework that, given a target security level for a specific application scenario, is able to i) discover the system elements, ii) quantify the security level of each element as well as its contribution to the security of the overall system, and iii) compute the control action to be applied on such elements to reach the security target. The main innovations proposed by the authors are: i) the definition of a comprehensive methodology to quantify the security of a generic system independently from the technology and the environment and ii) the integration of the derived metrics into a closed-loop scheme that allows real-time control of the system.
The solution described in this work moves from the proof-of-concepts performed in the early phase of the pSHIELD research and enriches it through an innovative metric with a sound foundation, able to potentially cope with any kind of application scenarios (railways, automotive, manufacturing, …).

Key-Words: - Complex Systems**,** Closed-loop, Dynamic Composability, E2E Security, Common Criteria, Attack surface metrics, Optimization

1 Introduction

Technological advances in computational capabilities along with improvement in communication technologies have enriched the market with a new class of SMART devices that can be used in every application domain, ranging from networking ([1]-[12]), automotive ([13]), multimedia ([14]-[16]), energy ([17]-[20]) or critical infrastructures protection ([21]-[23]).

These devices (i.e. sensor nodes, SMART actuators, programmable controllers, small computing platform, etc.) are commonly referred to as Embedded Devices or Embedded Systems (ESs) and their peculiarities are: i) a reduced size, ii) the possibility of implementing specific functionalities with limited resources and iii) the possibility of interconnecting with other devices to create more complex systems.

Leveraging these peculiarities, several industrial domains have started to massively deploy ESs networks to realize a plenty of tasks, no longer limited to a specific functionality but extended up to end-to-end behaviors.

In order to drive the European research towards an improvement of ES technologies, the European Commission, within the Seventh Framework Programme (FP7) has established the ARTEMIS-JU, a technological initiative in charge of defining and promoting a specific roadmap towards clear and focused objectives [24]. One of these objectives is the development of new technologies and/or strategies to address E2E Security in the context of ESs, with particular care to:

- Solutions oriented to systems certification,
- Cost reduction
- Re-use and re-engineering of non-recurring solutions.

In this context the authors, starting from their academic and industrial backgrounds, have conceived the SHIELD Framework ([25], [26]), an architectural paradigm and design methodology able to address security aspects potentially in each and every domain where ESs (or networks of interconnected ESs) are deployed to provide specific services.

As it happens for communication networks, where modular and cognitive architecture are adopted to provide flexible E2E services that dynamically satisfy the desired level of QoS (see [27]), similarly the interconnection of ESs may require the adoption of a modular and cognitive approach to provide E2E security functionalities that dynamically satisfy the desired "security level" form the end-user.

Thus, the main novelty of the presented approach is the possibility of realizing a "known and predictable" E2E security behaviour starting from the composition of individual, atomic elements. In spite of this, the main features of the proposed SHIELD framework are:

- **modularity** & **expandability** (i.e. the possibility of composing elements together),
- **cognitiveness** & **flexibility** (i.e. the possibility of dynamically adapting to the specific context)
- **technology independence** (i.e. the possibility of abstracting the controlled components in order to measure and provide security in any environment).

The composition problem for complex systems' control is not new in scientific landscape (see for example [29]), as well as the study of the security reliability properties of interconnected systems ([39], [42]) but what makes this problem really innovative in the application field chosen for this paper is that: i) the authors address the problem of controlling a property, i.e. security, that is not directly measurable in a deterministic way, so an hybrid qualitative/ quantitative approach is needed, and ii) the authors derive a technology-independent control scheme that can be tailored and applied on potentially any scenario.

The basic approach has already been presented in [25] as preliminary result of the pSHIELD research project [29]); in this paper an improvement with respect to the basic approach is shown, mainly basing on the recent advances achieved in the execution of the nSHIELD project ([30]), which represents the second phase of the SHIELD Roadmap. In particular this paper is the extended version of [31], describing more in detail the theoretical foundation of the new metric approach.

In order to describe the SHIELD approach to E2E security, the rest of the paper is structured as follows: in Section 2 the SHIELD methodology (as presented in [25]) is recalled and in Section 3 the SHIELD behaviour as a closed-loop control system is depicted in detail. In Section 4 the metric approach to measure E2E security is then presented in detail, and in Section 5 an example is provided. Finally in Section 6 some considerations on practical scheme implementation are reported, while in Section 7 conclusions are drawn.

2 The SHIELD methodology

The main purpose of the SHIELD methodology is to provide an architectural solution and a design paradigm to enable the Composability of atomic (Security) functionalities in Complex Systems.

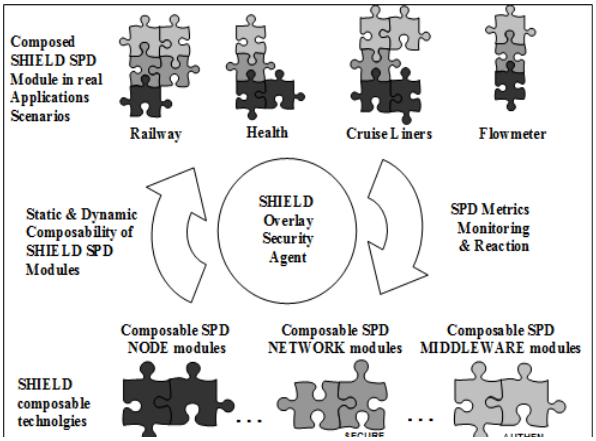

Fig. 1 SHIELD Methodology

A trivial representation is provided in Fig. 1. The *SHIELD modules* can be represented as pieces of a puzzle, which perfectly fits each other thanks to common interfaces. Each module implements a Security technology or a specific Security functionality. As an example, in Fig. 1 at *node level* there are two modules: Trusted Platform Module and Crypto Technology, at *network level* there are two functionalities: self-x algorithms and secure routing, and at *middleware level* there are two other services: semantic management and authentication.

These modules, belonging to different layers (node, network or middleware), can be composed (i.e. activated, deactivated or configured) statically or dynamically by the *SHIELD Security Agent*, an innovative software agent (see [1] for details) that collects the information on the system and takes decisions according to proper control algorithms.

This is possible thanks to the development of proper *semantic models* (as outlined in [32] and [33]) that allow the system description in a technology independent way (i.e. machine readable) as well as the definition of *security metrics* that allow the quantification of the security level.

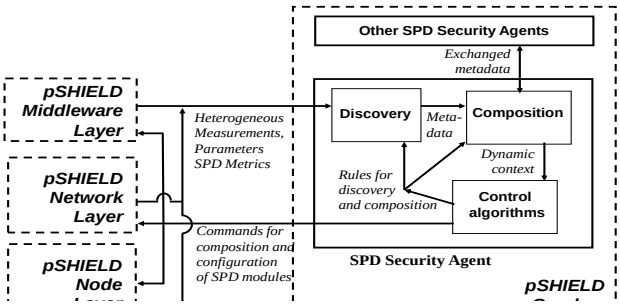

Fig. 2 SHIELD Architecture

Then, thanks to the continuous monitoring performed by the Security Agent, individual SHIELD modules can be dynamically activated and reconfigured once the measured Security metrics do not satisfy the required Security levels, even at run-time.

In addition modularity and technology-independence of the architecture allow a *plug&play-like* behaviour, suitable for any kind of application.

In a more structured representation, in Fig. 2 the SHIELD reference architecture is depicted as a control scheme, with the indication of the actors involved in the measurements and commands exchange. The scheme is generically referred to as SPD functionalities, that means Security Privacy and Dependability, since the proposed approach allows to jointly address these peculiarities. However in the prosecution of the paper we will refer only to the "Security" aspects, for which the new metrics and the control algorithms are tailored.

The core of the system, as previously introduced, is the Security Agent: each Agent monitors a set of properly selected measurements and parameters taken from the system (see the arrows labelled as *measurements* in Fig.2). These heterogeneous measurements and parameters are converted by the security agents in *homogeneous/technology-independent) metadata* by extensively using properly selected semantic technologies; the use of homogeneous metadata makes easy the metadata exchange among different security agent (see Fig.2).

Each Security Agent, thanks to metadata homogeneity, can aggregate the available metadata, in order to deduce information which form the so-called *dynamic context* on which the control decisions will be tailored.

Last, but not least, in the security agent runs a set of *control algorithms* which are responsible of dynamically deciding which Security modules have to be composed (i.e. enabled/ disabled/configured) in order to achieve the desired Security level. The decision is driven by the computation of proper technology independent *metrics*, specifically designed for security applications.

In the scope of control of security functionalities, that is the focus subject of this paper, the strength of the SHIELD methodology is that is possible to derive an overall (i.e. end-to-end) behaviour starting from the atomic behaviours of atomic components by composing them according to rules and control algorithms.

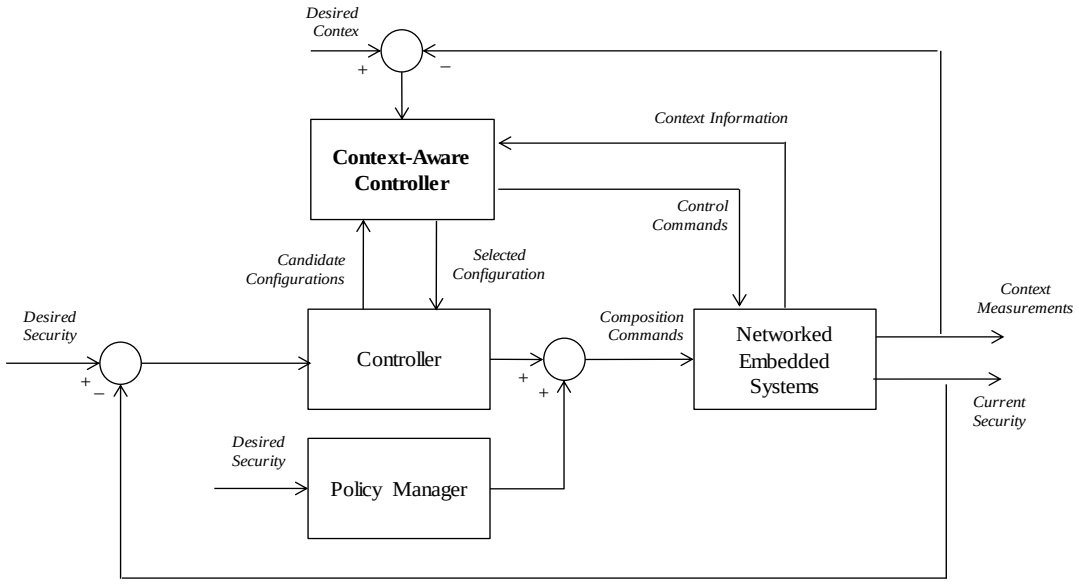

Fig. 3 Composability: a closed-loop view

On a practical point of view, the SHIELD paradigm allows to deploy small devices (or to use the ones already available), interconnect them and, with the introduction of an intelligent software Agent, dynamically organizing and structuring them so that their capabilities are leveraged to jointly produce the desired effect. As an example, one may be interested in realising the secure monitoring of a train station:

IF the devices deployed in the station (i.e. sensors, cameras, controllers, actuators, etc.) are SHIELD compliant

AND IF at least one SHIELD Security Agent is introduced in this system

THEN it is possible to activate the automatic composition and the system will automatically discover the available devices and the context information, quantify the security level according to the defined metrics, compute a control action and enforce it in the systems to activate/configure the sensors and cameras in the station so that the collected monitoring data are cyphered and made available only to authorized personnel.

This is a trivial example, but it is still representative of what is intended for E2E security behaviour: each component is in charge of a specific functionality that is useful to reach the overall objective.

3 The SHIELD closed-loop control approach

The problem of composing security functionalities can be successfully modelled by leveraging a control theoretic approach and deriving a closed-loop control scheme (see Fig.3). Indeed, such kind of model is by far closer to the effective implementation of the SHIELD system.

The *reference signal* is the desired security level, obtained and quantified according to the SHIELD metrics (that will be presented in the following section).

This signal is then used by the *Controller*, that is able to elaborate decision according to proper control algorithms as well as through the interaction with a secondary *Context Controller* that translates ancillary information on the system into constraints and parameters relevant for security purposes. A secondary reference signal may be applied to the system, if, apart from the E2E security behaviour, it is also of importance to control other parameters not relevant for security.

In [25] a control algorithm based on Common Criteria composition engine enriched with Hybrid Automata and Model Predictive Control optimization have been proposed as preliminary instantiation of such architecture. This approach has been conceived to be fully in line with the concepts being developed in similar context (e.g. the Future Internet framework [34], [35]) where the limitations coming from the lack of coordination among elements belonging to different layers and/or heterogeneous environments, are addressed through the design of modular controllers and multi-objective procedures.

This solution proved to be valid, but less effective for complex implementations mainly due to the effort needed to translate the "information" into semantic models. The nSHIELD research has

then lead to the definition of a new, simpler and more efficient approach, based on these pillars:

- A new metric has been introduced, based on the concept of "attack-surface", that enables an ease abstraction with respect to the underlying technologies.
- The Common Criteria (CC) guidelines have been confirmed, since the satisfaction of security properties must base its foundations on a consolidated standard, and embedded in the new metrics
- The control algorithm has become the translation of the metrics into an optimization problem, whose objective is to find the elements that maximize a target function

4 Metric approach to measure E2E security

In this section the authors introduce the basic SotA theory by which the SHIELD metric moves. This approach is entirely based on the *Open Source Security Testing Methodology* ([36]) OSSTMM which is a scientific methodology for the accurate characterization of "Operational Security" (OpSec) through examination and correlation of test results in a consistent and reliable way, and the *Attack Surface metrics* concept for software architecture ([38]). These theories are merged together to provide a solid basement and starting point for the paper's purposes. Then, this approach is improved through the application to non-software environments and enriched with procedural/ quantitative assessment derived from the Common Criteria evaluation methodology [20]. The main motivation is that approaching a security environment requires in any case to rely on consolidated standards, procedures or technologies as basic building blocks; the innovative contribution hereby presented is indeed given by the choice of these building blocks as well as their refinement to cope with context different from the software domain.

The result of this analysis-synthesis work is a metric that provides a self-consistent mean to translate "security properties" into a cardinal number that can be used to quantify the exposure of a system to threats; this number can be used to provide an intelligent controller (e.g. in a closed-loop scheme) with a reference signal that can be used to take decisions.

4.1 Attack-surface and OSSTMM metric

Since this paper merges two reference methodologies, some of the text addressing the basic definition and procedural steps in this section is entirely taken from the reference work [36] and [38]; changing the taxonomy would have indeed introduced mismatching in the methodology understanding to the reader.

To define the metric, we start with a basic statement: "*security measurement necessarily moves from the identification of what can cause damage to the system and to its dependability*". Typically, what causes damage is the "threat".

In fact by adopting a definition that integrates security and dependability concepts, one can assert that: "A *threat can be seen as the origin of the fault chain (fault->errors->failures-> damages) for the dependability concerns and as the potential for abuse of protected assets by the system for security concerns*". In this perspective, an attacker is the threat agent and it can be either a malicious human activity or a non-malicious event. How can the attacker affect the system? The answer is "through interactions": in fact the attacker uses the system's entry and exit points (i.e. its interfaces) to attack the system, so the identification of security threats basically involves identifying and controlling system's interfaces, being them internal or external.

The definition of the "attack-surface" is then given, as stated in [38] "*[…] intuitively, the attack surface of a system is the set of ways in which an adversary can enter the system and potentially cause damage […]".* In fact, according to the OSSTMM methodology, as first step of the metric computation procedure, an "entry and exit point framework" must be identified to perform security assessment and this entry and exit point framework can be assimilated to the "attack surface" that a system exposed to the external world.

On a practical point of view, this surface can be intended as a "virtual line" that surrounds a system and by which is possible to identify the potential menaces or vulnerabilities that affect the security level, i.e. its interfaces. The surface may dynamically evolve, so when new elements are inserted, removed or merged into the system, the attack surface is updated consequently and the new menaces/vulnerabilities are updated as well. Once the final shape of the system is achieved, the surface is frozen as the starting point to perform control. Potential control strategies that must be put in place to increase the security of the system should aim at reducing the attack surface, since it is reasonable that, as stated in [38] "*[…]the "smaller" is the attack surface, the more the system is secure[…]"*

To translate the surface into a manageable number, some numerical factors must be taken into account.

The first factor is the **porosity**, i.e. the grade of separation between the system and the attacker. Porosity is a function of three parameters:

- The "complexity", i.e. the number of components critical for the security, privacy and dependability of the system, which failure might not be tolerated by system architecture or another system within the scope. This positive integer value is indicated as P_C
- The "trust" level, i.e. the quantification of each relationship that exists where the system accepts interaction freely from its component. This positive integer value is indicated as P_T.
- The "access", i.e. the number of different places where the interaction (from outside the component/system) can occur. This positive integer value is indicated as P_A.

However, not all the "pores" identified through this last parameter are similar. For each access pore identified, it's necessary to introduce the concept of damage potential-effort ratio (DER) (as defined in [38]) to have a consistent measure of the lack of separation that introduces. DER is the ratio between the "Damage Potential" and the "Effort" values for each interface (1), and provides a numerical indicator of the damage that can be caused to the system if a malicious access occurs through the selected pore.

In fact the access pores don't contribute equally to system's porosity measurement because not all access pores are equally likely to be used by an attacker. A pore's contribution to a system's attack surface depends on the access pore's damage potential, i.e., the level of harm the attacker can cause to the system by using the access pore in an attack and the effort the attacker spends to acquire the necessary access rights in order to be able to use it in an attack. The higher the damage potential or the lower the effort, the higher access pore's contribution to the porosity, as indicated by the following formula:

$$\mathrm{DPE_i} = \frac{\mathrm{DP_i}}{\mathrm{DE_i}} \quad \forall \mathrm{i} \in \text{set of access} \tag{1}$$

We consider damage potential and effort in isolation while estimating a resource's contribution to the attack surface [38]. From an attacker's point of view, however, damage potential and effort are related; if the attacker gains higher privilege by using a method in an attack, then the attacker also gains the access rights of a larger set of methods. For example, the attacker can access only the methods with authenticated user access rights by gaining authenticated privilege, whereas the attacker can access methods with authenticated user and root access rights by gaining root privilege. The attacker might be willing to spend more effort to gain a higher privilege level that then enables the attacker to cause damage as well as gain more access rights. Hence we consider damage potential and effort in tandem and quantify a resource's contribution as a damage potential effort ratio. The ratio is similar to a cost-benefit ratio; the damage potential is the benefit to the attacker in using a resource in an attack and the effort is the cost to the attacker in using the resource.

Porosity access can then be further defined by the following formula:

$$\mathrm{P_A} = \sum_{\mathrm{i}} \mathrm{DPE_i} \; \forall \mathrm{i} \in \text{set of access} \tag{2}$$

Note that P_A is a natural number, so the value obtained by the previous formula is rounded to the nearest integer number (and it cannot assume negative values).

Then, starting from the above mentioned three basic values, the OSSTMM provides a numerical indicator to summarize the porosity value, named ***Operation Security*** (OpSec), that has been used in nSHIELD approach and is given by:

$$\mathrm{OpSec}_{sum} = a\mathrm{P_C} + b\mathrm{P_A} + c\,\mathrm{P_T} \tag{3}$$

Where a, b and c are may assume values in the interval [0; 1] to weight the individual contribution, according to environmental conditions of the system. OpSec_{sum} is a positive integer number so, as for P_A computation, the value obtained by the previous formula is rounded to the nearest integer number. This value can be expressed also in a normalized logarithmic form OpSec_{sumb}, as seen in the following.

The second factor to be considered in the OSSTMM metrics is the **control**.

Controls are a means to influence the impact of threats and their effects when interaction is required: they are, in a nutshell, the mean by which we can reduce the porosity of the system.

For the purposes of the SHIELD metrics, among all the possible controls, only ten controls categories have been selected from [36] as applicable. Controls categories can be grouped in two clusters.

The first cluster are the "Interactive Controls", i.e. those controls that directly influence complexity,

access, or trust interactions. They are (according to the official definitions in [36]) as follows:

- *Authentication* is a control through the challenge of credentials based on identification and authorization.
- *Indemnification* is a control through a contract between the asset owner and the interacting party. This contract may be in the form of a visible warning as a precursor to legal action if posted rules are not followed, specific, public legislative protection, or with a third-party assurance provider in case of damages like an insurance company.
- *Resilience* is a control over all interactions to maintain the protection of assets in the event of corruption or failure.
- *Subjugation* is a control assuring that interactions occur only according to defined processes. The asset owner defines how the interaction occurs which removes the freedom of choice but also the liability of loss from the interacting party.
- *Continuity* is a control over all interactions to maintain interactivity with assets in the event of corruption or failure.

The second cluster includes "Process Controls", i.e. those controls which are used to create defensive processes. These controls do not directly influence interactions rather they protect the assets once the threat is present. They are (according to the official definitions in [36]) as follows:

- *Non-repudiation* is a control which prevents the interacting party from denying its role in any interactivity.
- *Confidentiality* is a control for assuring an asset displayed or exchanged between interacting parties cannot be known outside of those parties.
- *Privacy* is a control for assuring the means of how an asset is accessed, displayed, or exchanged between parties cannot be known outside of those parties.
- *Integrity* is a control to assure that interacting parties know when assets and processes have changed.
- *Alarm* is a control to notify that an interaction is occurring or has occurred.

Three variables can be used to quantify the controls' effect:

- **LC**, i.e. Loss Control, that provides an indication of the Controls currently active/implemented in the system. In order to avoid disproportionate contributions of controls, the standard foresees the possibility of smoothing this value by means of logarithmic transformation, as follows. The normalized value is named **FC** (Full Control):

$$\mathrm{FC} = \log^2(1 + 10 \cdot \mathrm{LC}) \quad (4)$$

- **MC**, i.e. Missing Control, that represents the number of controls required to reach the balancing between the OpSec and the controls and is given by the following formula:

$$MC = \begin{cases} 0 & if\ OpSec_{sum} - LC \leq = 0 \\ \mathrm{OpSec_{sum}} - \mathrm{LC} & otherwise \end{cases} \quad (5)$$

- **TC**, i.e. True Control, that, in opposition to the Missing Controls, is used to measure the ideal placement of controls according to the formula:

$$\mathrm{TC} = \mathrm{OpSec_{sum}} - \mathrm{MC} \quad (6)$$

In order to avoid disproportionate placement of controls, the standard foresees the possibility of normalizing this value by means of logarithmic transformation, as follows:

$$\mathrm{TC_b} = \log^2(1 + 100(\mathrm{OpSec_{sum}} - \mathrm{MC} \cdot 0.1)) \quad (7)$$

Another useful instrument is the ***True Coverage*** (TCvg), expressing percentage ratio between controls and OpSec.

$$TCvg = \begin{cases} 0 & if\ OpSec_{sum} \leq = 0 \\ 1 - \frac{\mathrm{MC_{sum}}}{10 \cdot \mathrm{OpSec_{sum}}} & otherwise \end{cases} \quad (8)$$

As previously seen, True Control and Missing Control play a complementary role: it is easy to understand the ***Missing Coverage*** (MCvg).

$$\mathrm{MCvg} = 1 - \mathrm{TCvg} = \frac{\mathrm{MC_{sum}}}{10 \cdot \mathrm{OpSec_{sum}}} \quad (9)$$

The third and last factor in the overall metric definition is the **limitation**, that represents the incapacity of protection mechanisms to work properly; in other words it represents the holes, vulnerabilities, weaknesses, and any problem in keeping that separation between an asset and a threat or in assuring controls continue working correctly. The five Limitation classifications are (according to the official definitions in [36]) as follows:

- *Vulnerability* is the flaw or error that: (a) denies access to assets for authorized people or processes, (b) allows for privileged access to assets to unauthorized people or processes, or (c) allows unauthorized people or processes to hide assets or themselves within a defined scenario.
  This value can be indicated by $\mathbf{L_V}$ and must be weighted by a value that takes into account the

missing controls (that leads to vulnerability) as follows:

$$W_{L_V} = \frac{OpSec_{sum}+MC_{sum}}{OpSec_{sum}} \quad (10)$$

- *Weakness* is the flaw or error that disrupts, reduces, abuses, or nullifies specifically the effects of the five interactivity controls: authentication, indemnification, resilience, subjugation, and continuity.
This value can be indicated by $\mathbf{L_W}$ and must be weighted by a value that takes into account the missing Class A controls as follows:

$$W_{L_W} = \frac{OpSec_{sum}+MC_A}{OpSec_{sum}} \quad (11)$$

- *Concern* is the flaw or error that disrupts, reduces, abuses, or nullifies the effects of the flow or execution of the five process controls: non-repudiation, confidentiality, privacy, integrity, and alarm.
This value can be indicated by $\mathbf{L_C}$ and must be weighted by a value that takes into account the missing Class B controls as follows:

$$W_{L_C} = \frac{OpSec_{sum}+MC_B}{OpSec_{sum}} \quad (12)$$

- *Exposure* is an unjustifiable action, flaw, or error that provides direct or indirect visibility of critical assets for the security, privacy and dependability of the nSHIELD system within the chosen scenario interface.
This value can be indicated by $\mathbf{L_E}$ and must be weighted by a factor that takes into account complexity, accesses (scaled by the coverage percentage) plus the value of the limitation introduced by the other controls, as follows:

$$W_{L_E} = \frac{(P_V+P_A)\cdot MCvg+L_V+L_W+L_C}{OpSec_{sum}} \quad (13)$$

- *Anomaly* is any unidentifiable or unknown element which has not been controlled and cannot be accounted for in normal operations.
This value is indicated by $\mathbf{L_A}$ and is weighted by a factor that takes into account the trust values (i.e. major source of unforeseen anomalies) as well as the limitation introduces so far, according to the following formula:

$$W_{L_A} = \frac{P_T\cdot MCvg+L_V+L_W+L_C}{OpSec_{sum}} \quad (14)$$

The aggregated total limitation of the system is simply given by the weighted sum of these values. As previously seen, the evaluation can lead to two parameters: the first is an algebraic sum and the second is a base (logarithmic) form.

$$SecLim_{sum} = L_V \cdot W_{L_V} + L_W \cdot W_{L_W} + \\ + L_C \cdot W_{L_C} + L_E \cdot W_{L_E} + L_A \cdot W_{L_A} \quad (15)$$

$$SecLim_b = \log^2(1 + 100 \cdot SecLim_{sum}) \quad (16)$$

Controls and Limitations usually leads to a compromise. While controls are a positive influence in each possible SHIELD scenario, minimizing the attack surface, they can themselves add to the attack surface if they themselves have limitations. Often this effect is not noticed and if the protection mechanisms aren't tested thoroughly as to how they work under all conditions, this may not become evident. Therefore the use of controls must assure that they do not insinuate new attack interfaces into the target. Therefore, sometimes no controls is better than bad controls.
Fig. 4 shows how the Limitations are mapped with respect to the system and the controls.

Category		OpSec	Limitations
Operations		Visibility	Exposure
		Access	Vulnerability
		Trust	
Controls	Class A - Interactive	Authentication	Weakness
		Indemnification	
		Resilience	
		Subjugation	
		Continuity	
	Class B - Process	Non-Repudiation	Concern
		Confidentiality	
		Privacy	
		Integrity	
		Alarm	
			Anomalies

Fig. 4 Limitations effects

Finally, all these values can be put together to provide a quantification of the overall "security level" of the system (namely SEC). Each elementary value can be computed simply by counting the number of occurrences within the system (i.e. the number of limitation, the number of controls, …) while the other parameters are obtained through the presented formulas.

The first term to compute the final security value associated to the system is the ***SEC level Δ***.

$$\Delta_{SEC} = FC_b - OpSec_b - SecLim_b \quad (17)$$

The second term, namely ***True Protection*** is useful to understand the relationship among Porosity, True Control and Security Limitation.

$$\mathrm{TruPro} = 100 + \mathrm{TC_b} - \mathrm{OpSec_b} - \mathrm{SecLim_b} \quad (18)$$

The ***Actual SEC level*** is given by:

$$\mathrm{SEC} = 100 + \Delta_{\mathrm{SEC}} + -\frac{1}{100} \cdot (\mathrm{OpSec_b FC_b} - \mathrm{OpSec_b SecLim_b} + \mathrm{+FC_b SecLim_b}) \quad (19)$$

All the used values can be collected by a simple assessment of the system and/or be available in a system DB, thus making the security computation easy and at least semi-automatic.
As a conclusion, in [36] it has demonstrated that a system's attack surface measurement requires only the knowledge and computation of the system's factors porosity, controls, and limitation.

This represents the core of the SHIELD metric, improved with a Common Criteria methodology Injection.

4.2 Metric improvement by means of Common Criteria

Starting from what has been described in Section 4.1, the innovation brought by the authors and derived from the Common Criteria ([20]) is that the contribution provided by the limitations to the attack surface may vary from limitation to limitation and from one context to another, according to specific technological and environmental considerations.

This lead to the introduction of a function able to reflect such differences and quantify the impact of a limitation: this is done through the concept of "**Attack Potential**" described in the Common Criteria standard [20]. Attack potential is a function of expertise, resources and motivation. There are multiple methods for representing and quantifying these factors.

Motivation is an attack potential factor that describes several aspects related to the attacker and the assets the attacker desires to damage. At first, motivation can imply the likelihood of an attack (e.g. an attacker described as highly motivated means that an attack is imminent, or alternatively no attack is anticipated from an un-motivated attacker). However, except for the two extreme levels of motivation, it is difficult to derive a probability of an attack occurring from motivation.

Secondly, motivation can imply the value of the asset, monetarily or otherwise, either to the attacker or the asset holder. An asset of very high value is more likely to motivate an attack compared to an asset of little value. However, it is difficult to relate asset value to motivation because the value of an asset is subjective and it depends largely upon the value an asset holder places on it.

Last, but not least, motivation can imply the expertise and resources by which an attacker is willing to effect an attack. One can infer that a highly motivated attacker is likely to acquire sufficient expertise and resources to defeat the measures protecting an asset. Conversely, one can infer that an attacker with significant expertise and resources is not willing to effect an attack using them if the attacker's motivation is low.

Considering the second aspect, an asset holder may believe that the value of the assets (however measured) is sufficient to motivate attack against them. The attacker's motivation is considered to determine the methods of attack that may be attempted, as well as the expertise and resources used in those attacks.

Expertise and resources reflect the tools used by the attacker to perform the attack. Tab. 1 identifies the factors used to compute the *Attack Potential* and the associated numeric values with the total value of each factor.

Where a factor falls close to the boundary of a range the analyst should consider use of an intermediate value to those in the table. For example, if twenty samples are required to perform the attack then a value between one and four may be selected for that factor, or if the design is based on a publicly available design but the developer has made some alterations then a value between zero and three should be selected according to the analyst's view of the impact of those design changes. The table is intended as a guide.

Factor	Value
Elapsed Time	
<= one day	0
<= one week	1
<= two weeks	2
<= one month	4
<= two months	7
<= three months	10
<= four months	13
<= five months	15
<= six months	17
> six months	19
Expertise	
Layman	0
Proficient	3
Expert	6
Multiple experts	8

Factor	Value
Knowledge of Target of the attack	
Public	0
Restricted	3
Sensitive	7
Critical	11
Window of Opportunity	
Unnecessary/ unlimited access	0
Easy	1
Moderate	4
Difficult	10
None	**
Equipment	
Standard	0
Specialized	4
Bespoke	7
Multiple bespoke	9

Tab. 1 Attack Potential Computation parameters

The "**" specification in the table in considering Window of Opportunity is not to be seen as a natural progression from the timescales specified in the preceding ranges associated with this factor. This specification identifies that for a particular reason the potential vulnerability cannot be exploited in the target of the attack in its intended operational environment. For example, access to the target of the attack may be detected after a certain amount of time in a known environment (i.e. in the case of a system) where regular patrols are completed, and the attacker could not gain access to the target of the attack for the required two weeks undetected. However, this would not be applicable to a target connected to the network where remote access is possible, or where the physical environment of the target of the attack is unknown.

In order to determine the resistance of the target of the attack to the identified potential vulnerabilities, the following steps shall be applied:

a) Define the possible attack scenarios {AS1, AS2... ASn} for the target of the attack in the operational environment.

b) For each attack scenario, perform a theoretical analysis and calculate the relevant attack potential using Tab. 1.

c) For each attack scenario, if necessary, perform penetration tests in order to confirm or to disprove the theoretical analysis.

The "Values" column of Tab. 2 indicates the range of attack potential values (calculated using Tab. 1) of an attack scenario that results in the SPD functionalities being undermined.

Values	Attack potential required to exploit scenario	Target resistant to attackers with attack potential of
0-9	Basic	No rating
10-13	Enhanced-Basic	Basic
14-19	Moderate	Enhanced-Basic
20-24	High	Moderate
=>25	Beyond High	High

Tab. 2 Rating of vulnerabilities and attacked target resistance

Such kind of approach cannot take into account each and every circumstance or factor, but should give a better indication of the level of resistance to attack required to achieve the standard ratings. Other factors, such as the reliance on unlikely chance occurrences, are not included in the basic model but can be used by an analyst as justification for a rating other than those that the basic model might indicate.

It should be noted that whereas a number of vulnerabilities rated individually may indicate high resistance to attack, collectively the combination of vulnerabilities may indicate that overall a lower rating is applicable. The presence of one vulnerability may make another easier to exploit.

In the SPD level computation, the value assigned to vulnerabilities is calculated by performing a weighted sum of all identified vulnerabilities categorized according to the Tab. 2. The weight for each category of vulnerability (basic, enhanced basic, moderate, high and beyond high) is assigned by constant values (empirical parameters derived from experience) that are characterized by being inversely proportional to the attack potential required to exploit the scenario; this is because the lower the attack potential required to exploit the scenario the greater the impact of the vulnerability and then the attack surface on which is based the whole theory. For further reference on this computation approach, please refers to SHIELD project's documents ([45]).

4.3 Metric Composition rules

Another key innovation of the SHIELD metrics is their **scalability** with respect to the number of systems' elements. In fact the metric computation is easy and can be applied to individual components as well as to group of components in the same way, by simply applying some basic composition rules.
More in detail:

- Step 1: starting on system architecture, it is necessary to define by successive steps how all the various components involved in the overall computation are physically connected to create complex elements and/or sub-systems. The output is coupling sequence (for example if there are five elements A, B, C, D and E, the result may be A is connected with B, C is connected with D and E, so the final coupling sequence will be $(A \cup B) \cup (C \cup D \cup E)$).
- Step 2: following the coupling sequence, the basic parameters for metric computation (equations from (1) to (16)) must be put together by composing the corresponding values (for example the complexity values of two components is summed, the number of controls is summed for each and every category, and so on). Composition rules are reported in the following
- Step 3: finally the Actual Security Level (equations from (17) to (19)) is computed

only once, at the end of the procedure, on the basis of the parameters derived in Step 2.

The rules of metrics composition used to couple basic parameters are as follows:

- **Complexity** ***(SUM)***
  All critical elements whose failure might not be tolerated by system architecture must be considered. (If the same element is critical for more than one component it must be considered only once; if a single component has more than one critical element, it must be considered as 1 in the composition)
- **Access** ***(SUM for the different types)***
  All possible accesses to the composition of components must be considered. (If one access is common to both components, it must be considered as 1; if one access of the first component belongs also to the other component and is internal to the composition of components (with a relationship of trust) then these accesses must not be considered and must be augmented of one unit the Trust element)
- **Trust** ***(SUM)***
  Each relation that exists wherever the system accepts interaction freely from its component or another system within the scope, must be considered.
- **Confidentiality, Privacy, Authentication, Resilience, Integrity, Non-repudiation, Subjugation, Continuity, Indemnification, Alarm** ***(SUM for the different categories)***
  All controls that counteract threats and their effects must be considered.
- **Exposure** ***(SUM)***
  All unjustifiable action, flaw, or error that provides direct or indirect visibility of targets or assets within the chosen scenario interface must be considered. (If the same element represent an exposure for more than one component it must be considered only once)
- **Vulnerability** ***(SUM for the different rating)***
  All possible flaw or error that: (a) denies access to assets for authorized people or processes, (b) allows for privileged access to assets to unauthorized people or processes, or (c) allows unauthorized people or processes to hide assets or themselves within the scope, must be considered.
- **Weakness** ***(SUM for the different types)***
  All possible flaws or errors that disrupts, reduces, abuses, or nullifies specifically the effects of the five interactivity controls: authentication, indemnification, resilience, subjugation, and continuity must be considered.
- **Concern** ***(SUM)***
  All possible flaws or errors that disrupts, reduces, abuses, or nullifies the effects of the flow or execution of the five process controls: non-repudiation, confidentiality, privacy, integrity, and alarm, must be considered
- **Anomaly** ***(SUM)***
  All unidentifiable or unknown elements which cannot be accounted for in normal operations, generally when the source or destination of the element cannot be understood, must be considered. (If more than one component considers the same anomaly it must be counted only once)

The final procedure to compute SPD level for a generic complex system, starting from its individual components, is reported in Fig. 5 and represents one of the SHIELD projects breakthrough.

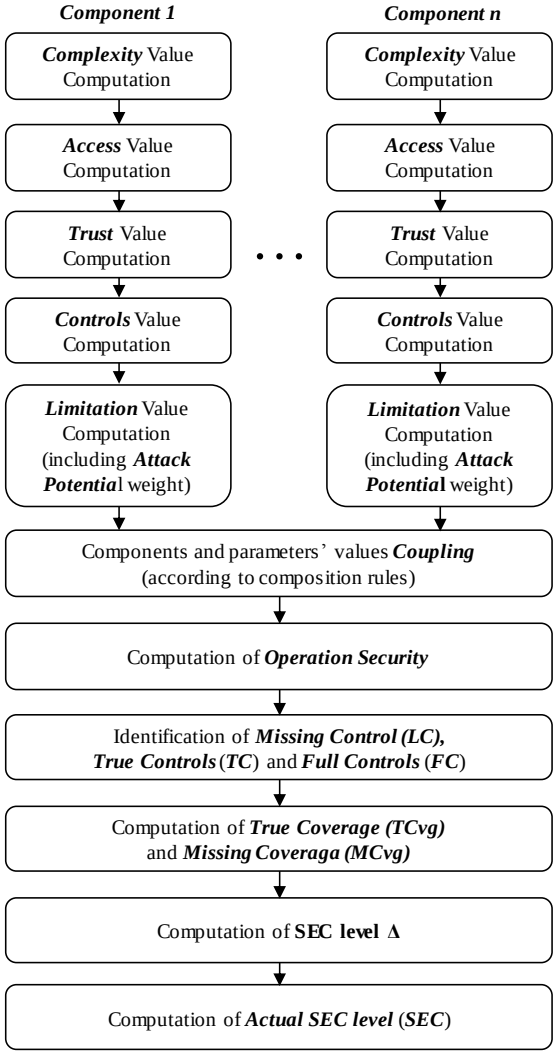

Fig. 6 Metrics Computation procedure

4.3 SHIELD closed-loop (metric-based) control scheme

Once the attack surface is computed according to the procedure defined so far, a control algorithm can be put in place to use these metrics to drive the composability problem of security functionalities.

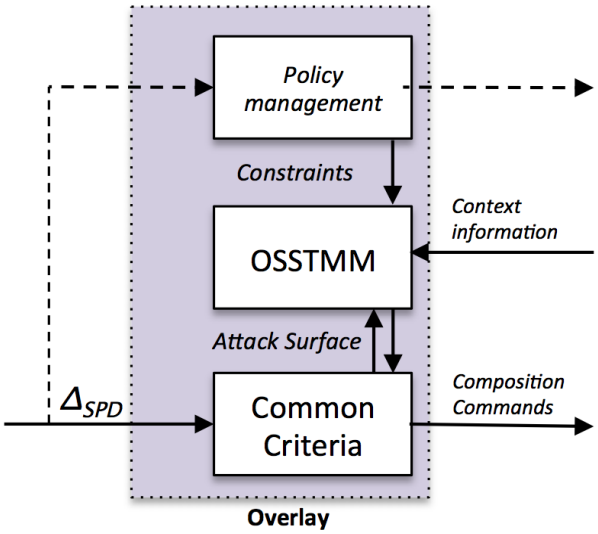

Fig. 6 Innovative Controller

The result is depicted in Fig. 6:

- The Context Aware controller, relying on the OSSTMM standard, uses context information to provide the list of Interactive/Process Controls that the main controller may put in place to cope with the security needs.
- The main controller takes the current value of the metric plus the set of functionalities available in the system and, basing on a look-up table or an optimization function, tries to minimize the vulnerability of the attack surface by activating functionalities and inserting "controls".

To make this scheme more efficient, the influence of a Policy Management (that in [25] has been modelled as a disturb) is considered and hereby modelled has a "controllable" input for the context controller, that considers Policies as constraints to the Interactive/Process controls to put in place.

From a mathematical point of view, the main controller solves a typical optimal control problem (e.g. [41]) where:

- the objective function is the minimization/ maximization of the Security value associated to a specific system configuration
- the constraints is given by the OSSTMM-CC standards (and by the policy management system).
- the candidate solutions are all the potential configurations of systems elements (or controls to be activated), each of them associated to a specific metric computed as described above.

In a nutshell, maximizing the Security level cause the activation of more controls and countermeasures, i.e. the choice of a system configuration with the highest metric value. In particular, this is achieved by minimizing the Δ_{SEC} (i.e. security) value, which is the difference between the desired and actual SEC values. It must be noted that the problem may be solve either by enumerating all the possible elements configuration and sorting them, or by building a more compact formulation (e.g. linear programming) and solving it by means of classical approaches.

In this paper a specific formulation of the optimization problem is not provided, since the main purpose is to present the theoretical frameworks, as well as the innovative metric for security that is decoupled from the mathematical theory adopted to solve it; in fact other instruments, different from simple optimization, can be adopted, like for example Petri Nets ([43]): a clear example of the application of Petri Nets to composable security is given in [44].

This framework opens indeed the way to several improvements and enrichment on the "solving" part.

5 Example of the SHIELD approach

The example by which the proposed methodology has been tested is an improvement of the one presented in [25] as final demonstration of the pSHIELD project, i.e. the "Monitoring of freight trains transporting hazardous material".

The hypothesized platform is composed by a central unit connected by means of a ciphered wireless network to remote sensors. In this platform the assets to protect are data sent by remote sensors to central unit, where data are recorded inside the central unit itself.

Threats identified for the above scenario are the following:

- Unauthorized disclosure of information stored within or communicated through computers or communications systems;
- Unauthorized modification or destruction of stored information;
- Manipulation of computer or telecommunications services resulting in various violations;
- Propagation of false or misleading information;
- Users lacking guidance or security awareness;

- Data entry or utilization error;
- Faulty access rights management;

Security functionalities (i.e. Controls) that counteract the above mentioned threats belong to the following categories:

- Authentication;
- Confidentiality;
- Non repudiation;
- Subjugation.

The application of the surface Attach metrics approach does not depend on a thorough knowledge of the theory that generates such an approach, but only by a well-established knowledge that the supplier of the system and/or components of a system must have on security issues.

Starting from the previously evidenced threats, for each of the two components the attack surface value must be computed, according to the guidelines provided in [20] and [36].

The values for the components of the sample scenario are:

- Central unit: 88,75 (constant due to the lack of controls that could be implemented)
- Wireless Sensor Network: [84,089 93,340]

Depending on which of the two available controls is activated. In fact it is important to consider that the different choice of key management and Cryptographic operation algorithm change the vulnerability type, so it insert the possibility, changing these algorithm to modify the Security level of the component introducing different states.
In this case the formulation of an Optimization function is not needed, since it is evident that the most robust configuration is the one associated to a 93,340 value for the WSN. However, in case the available controls and their combination is very high, it is sufficient to maximize the Optimization function given by the sum of the atomic security value, within the constraints defined by policies (i.e. mutual inclusion or mutual exclusions of controls)

6 Control scheme implementation

In the SHIELD roadmap demonstrators ([29] and [30]) a sample implementation of the presented control architecture has been proposed and tested on real application scenarios. Since the SHIELD framework is thought to be deployed on large scale complex systems composed by dozens or hundreds of devices, it seems impossible to think about pervasive HW/SW injection in the system; in other words, such solution, to be effective, must be seamless deployable into an existing system, in order not to impact the current operational capabilities. This objective has been achieved in two ways:

- By minimizing the number of new HW equipment to be put in place
- By minimizing the SW upgrade to be applied to the system

The first point is immediately addressed with the architectural constraints described in Section2: the proposed methodology in fact requires the presence of at least (only) one entity, named Security Agent, to run into the system to collect/compute metrics.

This software entity can be installed into existing nodes with high computational capabilities (i.e. no need of HW injection); if no powerful nodes are available, then (only) one new node can be inserted, in any system entry point. This high capability node is requested to run common Service Discovery Protocols to collect the metrics from the different devices: such protocols are freeware and a plenty of libraries have been developed for the most common languages. The Security Agent implements also the control algorithm, that can be either a look-up table or an optimization function, whose libraries are freeware as well.

To address the second need (minimize SW upgrade) we can assert that no upgrade is needed in the rest of the system apart from the presence of a module able to reply to the discovery requests: this capability is native in most common devices, so the assertion in reasonable. This discovery answering module is indeed responsible of capturing the discovery requests coming from the security agent and answering with the list of parameters used to compute the metrics (as indicated in Section 4).

Last, but not least, the key enabling technology that makes the system work is Ontology, i.e. a tool to represent information in a structured way (see [32] and [40] as example). A proper SHIELD ontology has been designed to include all the OSSTMM, attack surface and Common Criteria related parameters, to that the information exchange between the system's elements is efficient and requires basic network connectivity. This increase the feasibility of SHIELD deployment over systems with dozens of components.

7 Conclusion

In this paper the innovative results achieved by the nSHIELD project have been presented, as a significant improvement of the proof of concept reported in [25], with additional details with respect

to the short version of this work ([31]). In particular it has been shown how it is possible to drive an E2E behaviour by acting on the atomic elements; the key idea is to describe each component with a clear and univocally defined metric value that measure the vulnerability of its attack surface (derived as a mix of [20], [36] and [38] guidelines). Then, while composing together several elements, the resulting attack surface is obtained as the result of an optimization problem whose potential solutions are the different controls that the atomic elements can put in place to countermeasure specific menaces. The problem may be solved by exploration or through simple heuristics.

The proposed methodology is currently being intensively tested in industrially relevant scenarios from the avionic and railways domains and the results will be made available in the final nSHIELD project deliverables.

Future works foresee the adaptation of the proposed approach to address also other problems. It could be particularly helpful, for example, in scenarios where the topologies change very often and the E2E behavior is the provisioning of a specific service, like power distribution (see [37]). The main challenge will be the adaptation/tailoring of a proper metric to the new domain, since a good metric is the basis of any SHIELD-like methodology.

Acknowledgements

The authors would like to thank the pSHIELD and nSHIELD partners (in particular, V. Suraci for his valuable contributions) for the significant work performed in these projects to transfer the SHIELD methodology from idea to reality.
An acknowledgement goes also to the authors of [36] and [38] for their valuable contribution to the state of the art with a theory that proved to be effective for the SHIELD needs and that has been entirely endorsed, and a bit refined, to fit the selected problem

References:

[1] A. Palo, L. Zuccaro, A. Simeoni, V. Suraci, L. Musto, P. Garino, "A common open interface to programmatically control and supervise open networks in the future internet," in *2013 Future Network and Mobile Summit*, pp. 1 – 9, Lisbon, July 2013. ISBN: 978-190582437-3.

[2] G. Oddi, M. Panfili, A. Pietrabissa, L. Zuccaro, V. Suraci, "A resource allocation algorithm of multi-cloud resources based on Markov Decision Process," in *IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom 2013)*, Bristol, UK, 2-5 December, 2013, pp. 130-135, doi: 10.1109/CloudCom.2013.24.

[3] T. Meyer, V. Suraci, P. Langendörfer, S. Nowak, M. Bahr, R. Jennen, "An inter-MAC architecture for heterogeneous Gigabit home networks," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, pp. , vol. 5449931, Tokio, 2009. ISBN: 978-142445123-4. DOI: 10.1109/PIMRC.2009.5449931

[4] M. Castrucci, M. Cecchi, F. Delli Priscoli, L. Fogliati, P. Garino, V. Suraci, "Key Concepts for the Future Internet Architecture," in *2011 Future Network & Mobile Summit*, Warsaw, June 2011. ISBN: 9781905824236.

[5] Mignanti S, Di Giorgio A, Suraci V (2008). A Model Based RL Admission Control Algorithm for Next Generation Networks. In: AlBegain K; Cuevas A. Proceedings - The 2nd International Conference on Next Generation Mobile Applications, Services, and Technologies, NGMAST 2008. Cardiff, WALES, SEP 16-19, 2008, ISBN: 978-0-7695-3333-9, doi: 10.1109/NGMAST.2008.19

[6] G. Oddi A. Pietrabissa,, F. Delli Priscoli, V. Suraci, "A decentralized load balancing algorithm for heterogeneous wireless access networks", in *World Telecommunication Congress (WTC)*, Berlin, June 2014.

[7] V. Suraci, V. Castrucci, G. Oddi, A. Cimmino, R. Colella, "Convergence in Home Gigabit Networks: Implementation of the Inter-MAC Layer as a Pluggable Kernel Module," in *21st Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communication*, pp. 2569 – 2574, Instanbul, September 2010. ISBN: 978-1-4244-8017-3. DOI: 10.1109/PIMRC.2010.5671761.

[8] G. Oddi, A. Pietrabissa and F. Liberati, "Energy balancing in multi-hop Wireless Sensor Networks: an approach based on reinforcement learning", in *Proc. of 2014 NASA/ESA Conference on Adaptive Hardware and Systems (AHS-2014)*, Leicester, UK.

[9] G. Oddi, A. Pietrabissa, "A distributed multi-path algorithm for wireless ad-hoc networks based on Wardrop routing". in *Proc. of the 21st Mediterranean Conference on Control & Automation (MED)*, 2013 , doi: 10.1109/MED.2013.6608833.

[10] A. Pietrabissa, "Admission Control in UMTS Networks based on Approximate Dynamic Programming," *European Journal of Control*,

vol. 14, no. 1, pp. 62-75 , January 2008, doi:10.3166/ejc.14.62-75.

[11] A. Pietrabissa, "An Alternative LP Formulation of the Admission Control Problem in Multi-Class Networks," *IEEE Transaction on Automatic Control*, vol. 53, no. 3, pp. 839-845, April 2008, doi: 10.1109/TAC.2008.919516.

[12] A. Pietrabissa, "A Reinforcement Learning Approach to Call Admission and Call Dropping Control in Links with Variable Capacity", *European Journal of Control*, vol. 17, no. 1, pp. 89-103, 2011, ISSN 0974-3580, DOI: 10.3166/EJC.17.89-103.

[13] A. Di Giorgio, F. Liberati, S. Canale, "IEC 61851 compliant Electric vehicle charging control in Smartgrids," in *21st Mediterranean Conference on Control and Automation MED13, IEEE*, pp. 1329 – 1335, Chania, GR, June 2013. DOI: 10.1109/MED.2013.6608892.

[14] V. Suraci, A. Palo A, "Green networking: Integration of Wireless Sensor Network with Home Gigabit Access," in *2011 Future Network & Mobile Summit 2011*, Warsaw, June 2011. ISBN: 978-1-905824-16-8.

[15] F. Delli Priscoli, "Dynamic Class of Service Mapping for Quality of Experience Control in Future Networks", in *World Telecommunication Congress (WTC)*, Berlin, June 2014.

[16] V. Suraci, S. Mignanti, A. Aiuto, "Context-aware Semantic Service Discovery" in *Proocedings of Mobile and Wireless Communications Summit*, pp. 1 – 5, Budapest, July 2007. ISBN: 963-8111-66-6. DOI: 10.1109/ISTMWC.2007.4299110

[17] A. Mercurio, A. Di Giorgio, P. Cioci, "Open Source Implementation of Monitoring and Controlling Services for EMS/SCADA Systems by Means of Web Services," *IEEE Transactions on Power Delivery*, vol. 24, no. 3, pp. 1148-1153, 2009. DOI: 10.1109/TPWRD.2008.2008461.

[18] A. Di Giorgio A., L. Pimpinella, F. Liberati, "A model predictive control approach to the load shifting problem in a household equipped with an energy storage unit," in *20st Mediterranean Conference on Control and Automation MED12, IEEE*, pp. 1491 – 1498, Barcelona, ES, July 2012. DOI: 10.1109/MED.2012.6265850.

[19] V. Suraci, A. Marucci, R. Bedini, L. Zuccaro, A. Palo, "Energy-aware control of home networks," in *Lecture Notes in Computer Science*, pp. 299 – 311, vol. 7972, 2013. ISBN: 978-364239642-7. DOI: 10.1007/978-3-642-39643-4_23.

[20] Common Criteria for Information Technology Security Evaluation, v3.1, July 2009. Available: www.commoncriteriaportal.org.

[21] A. Di Giorgio, F. Liberati, "Interdependency modeling and analysis of critical infrastructures based on Dynamic Bayesian Networks," in *19st Mediterranean Conference on Control and Automation MED11*, *IEEE*, pp. 791–797, Corfu, GR, June 2011. DOI: 10.1109/MED.2011.5983016.

[22] P. Capodieci, S. Diblasi, E. Ciancamerla, M. Minichino, C. Foglietta, D. Lefevre, G. Oliva, S. Panzieri, R. Setola, S. De Porcellinis, F. Delli Priscoli, M. Castrucci, V. Suraci, L. Lev, Y. Shneck, D. Khadraoui, J. Aubert, S. Iassinovski, J. Jiang, P. Simoes, F. Caldeira, A. Spronska, C. Harpes, M. Aubigny, "Improving Resilience of Interdependent Critical Infrastructures via an On-Line Alerting System," in *Complexity in Engineering, 2010. COMPENG'10*, p. 88 – 90, Rome, IT, Feb 2010. ISBN: 978-1-4244-5982-7

[23] F. Caldeira, M. Castrucci, M. Aubigny, D. Macone, E. Monteiro, F. Rente, P. Simoes, V. Suraci, "Secure Mediation Gateway Architecture Enabling the Communication Among Critical Infrastructures," in *2010 Future Network and Mobile Summit*, pp. 1 – 9, Florence, IT, June 2010. ISBN: 978-1-905824-16-8

[24] ARTEMIS Strategic Research Agenda, March 2006. Available: www.artemis-ia.eu

[25] F. Delli Priscoli, A. Fiaschetti, V. Suraci, "The SHIELD Framework: how to control Security, Privacy and Dependability in Complex Systems," in *IEEE Workshop on Complexity in Engineering*, pp. 1-4, Aachen, June 2012. DOI: 10.1109/CompEng.2012.6242962

[26] V. Suraci, A. Fiaschetti, G. Anzidei, "Design and implementation of a service discovery and composition framework for security, privacy and dependability control," in *Future Network & Mobile Summit 2012*, Berlin, DE, July 2012.

[27] M. Castrucci, F. Delli Priscoli, A. Pietrabissa, V. Suraci, "A cognitive future internet architecture," (2011) Lecture Notes in *Computer Science*, pp. 91 – 102, vol. 6656. ISBN: 978-364220897-3. DOI: 10.1007/978-3-642-20898-0_7

[28] Armen G. Bagdasaryan, "System Approach to Synthesis, Modeling and Control of Complex Dynamical Systems" in WSEAS Transactions on Systems and Control, Issue 2, Volume 4,

February 2009, ISSN: 1991-8763, E-ISSN: 2224-2856
[29] pSHIELD Technical Annex, June 2010.
[30] nSHIELD Technical Annex, September 2011.
[31] A. Fiaschetti, A. Morgagni, A. Lanna, M. Panfili, S. Mignanti, R. Cusani, G. Scarano, A. Pietrabissa F. Delli Priscoli "Control Architecture to Provide E2E security in Interconnected Systems: the (new) SHIELD Approach" in *Proceedings Santorini* ISBN 978-1-61804-237-8
[32] A. Fiaschetti, V. Suraci, F. Delli Priscoli, A. Taglialatela, "Semantic technologies to model and control the "composability" of complex systems: a case study," *Book chapter of Horizons in Computer Science Research*, Nova Publisher, 2012.
[33] A. Fiaschetti, F. Lavorato, V. Suraci, A. Palo, A. Taglialatela, A. Morgagni, R. Baldelli, F. Flammini F., "On the use of semantic technologies to model and control security, privacy and dependability in complex systems," in *Computer Safety, Reliability, and Security*, pp. 467 – 479, Springer, 2011. DOI: 10.1007/978-3-642-24270-0_34.
[34] F. Delli Priscoli, "A fully cognitive approach for future internet," in *Future Internet*, vol. 2, no. 1, pp. 16–29, 2010.
[35] F. Delli Priscoli, A. Pietrabissa, V. Suraci, M. Iannone, "Modelling Quality of Experience in Future Internet Networks," in *2012 Future Network & Mobile Summit*, pp. 1 – 9, Berlin, DE, 2012. ISBN: 9781467303200.
[36] OSSTMM, Open Source Security Testing Methodology Manual. Available: www.osstmm.org
[37] S. Canale, A. Di Giorgio, A. Lanna, A. Mercurio, M. Panfili, and A. Pietrabissa, "Optimal planning and routing in medium voltage powerline communications networks," in *IEEE Trans. Smart Grid*, vol. 4, no. 2, pp. 711–719, 2013.
[38] Pratyusa K. Manadhata and Jeannette M. Wing "An Attack Surface Metric" in *IEEE Transactions on Software Engineering*, 2010
[39] Kirti Tyagi and Arun Sharma, "Reliability of Component Based Systems – A Critical Survey" *in WSEAS Transactions on Computers*, Issue 2, Volume 11, February 2012, E-ISSN: 2224-2872
[40] Hema M. S. and Chandramathi S., "Quality Aware Service Oriented Ontology Based Data Integration", in *WSEAS Transactions on Computers*, Issue 12, Volume 12, December 2013, E-ISSN: 2224-2872
[41] Zemliak A., Pena R. and Rios E., "Generalized Optimization Methodology for System Design" in *WSEAS Transactions on Systems*, Issue 6, Volume 9, June 2010, E-ISSN: 1109-2777
[42] Kasem M. and Chahin N., "A Novel Approach for Meeting the Challenges of the Integrated Security Systems", in *WSEAS Transactions on Systems*, Issue 9, Volume 10, September 2011, E-ISSN: 1109-2777
[43] Cem B. and Kurtulan S., "Generalized State Equation for Petri Nets", in *WSEAS Transactions on Systems*, Issue 9, Volume 10, September 2011, E-ISSN: 1109-277
[44] nSHIELD Consortium, "Deliverable D5.2 Preliminary Middleware and Overlay Prototypes", in *nSHIELD Project*, 2012
[45] nSHIELD Consortium, "Deliverable D2.8 Final nSHIELD Metrics", in *nSHIELD Project*, 2013