A. Nieto, and J. Lopez, "Contextualising Heterogeneous Information in Unified Communications with Security Restrictions", *Computer Communications*, vol. 68, pp. 33-46, 2015.
http://doi.org/10.1016/j.comcom.2015.07.015
NICS Lab. Publications: https://www.nics.uma.es/publications

Contextualising Heterogeneous Information in Unified Communications with Security Restrictions

Ana Nieto[a], Javier Lopez[a]

[a] *Computer Science Department, University of Malaga, Ada Byron building, 29071 Malaga, SP.*

Abstract

The lack of abstraction in a growing semantic, virtual and abstract world poses new challenges for assessing security and QoS tradeoffs. For example, in Future Internet scenarios, where Unified Communications (UC) will take place, being able to predict the final devices that will form the network is not always possible. Without this information the analysis of the security and QoS tradeoff can only be based on partial information to be completed when more information about the environment is available. In this paper, we extend the description of context-based parametric relationship model, providing a tool for assessing the security and QoS tradeoff (SQT) based on interchangeable contexts. Our approach is able to use the heterogeneous information produced by scenarios where UC is present.

Keywords: Security; QoS; tradeoff; Context;

1. Introduction

Future Networks are intended to be convergent networks, where multi-purpose devices can coexist and cooperate to achieve common goals, or even work in an oportunistic symbiosis. Therefore, *Unified Communications* (UC) play an important role by opening the door of the collaboration between heterogeneous devices in different environments [1]. This entails several challenges, for example, what happens to the valuable information generated by the collaboration, how it can be used and also protected and, moreover, how to handle the different environments to be analysed to identify the dependencies between the parameters that may be critical (e.g., for identifying cascade effects). Furthermore, in *Future Internet* (FI) scenarios, where multiple devices coexist and networks are always changing, performing tests or simulations before the deployment of the solutions is very complex, because the number of factors to be considered and measured increase exponentially. We are especially concerned about security and Quality of Service (QoS) mechanisms, because both are fundamental in providing a total network convergence [2].

Some of the challenges regarding the security and QoS are motivated by the users' growing demands, the unification of heterogeneous and dynamic environments, and the problems in predicting the behaviour of the final ecosystem. Users' applications and services necessarily require the existence of both, QoS and security mechanisms [3, 4], in order to guarantee, at least to some degree, that the network will be able to satisfy the user's demand for performance (e.g., delay, response time) and security (e.g., privacy, integrity, confidentiality). Furthermore, the capability of providing QoS or security mechanisms also depends on the resources of the environment (e.g., bandwidth, coverage provided by the antennas) and the resources of the devices (e.g., memory, security architecture). In general, not all the mechanisms can coexist in the same environment without affecting each other.

Moreover, the user's mobility between different domains complicates the traceability of malicious devices, and poses serious challenges to the resource provisioning, that are based on the predictability of being efficient. In this respect, in [5] the authors provide an interesting map and classification of where the data in cellular networks is located and generated. This information is not only personal, but also includes measurements about the perfor-

Email addresses: nieto@lcc.uma.es (Ana Nieto), jlm@lcc.uma.es (Javier Lopez)

mance of the mechanisms in the network which can be stored and analysed to improve the configuration of the systems and to predict when the peaks of demand of resources occur.

Therefore, as more systems converge, more information is available to understand the behaviour of the network and to be used to analyse the security and QoS tradeoff [6]. This analysis must to be done considering partial information, because the dynamic nature of these environments reveals the requirements, properties and characteristics to be considered, but not the final mechanisms that must be available for implementing the security and QoS requirements. For example, in ad-hoc communications, the solutions to be deployed depend a lot on the capabilities of the devices. If the devices do not support the communication protocol, then diverse security and QoS mechanisms cannot be applied [7]. Moreover, as the user is included as part of the network, and is able to interact with the things or objects around him/her, new solutions should be deployed and configured considering a wide range of purposes in mind, based on a context, where subjective values have to be considered. The subjectivity of a component is essential, to identify the components or things that are fundamental to a user, or any other actuator in the network, at any given moment.

We define the *generic models for assessing the security and QoS tradeoff* as those models capable of analysing the security and QoS requirements and characteristics of a set of elements and components in a system. These models are capable of changing the composition of these elements and characteristics for others and still be useful. The idea behind these types of models is that a part of the model has to remain abstract prior to knowing or receiving the new components in the information system. So, these models are well suited to use in heterogeneous networks of dynamic composition, as is the case of UC in FI environments, where it is very difficult to predict with any great accuracy the devices that will comprise the network.

In this paper we define a tool for assessing the *Security and QoS tradeoff* (SQT) in heterogeneous networks, taking as the basis, a set of well defined security and QoS parameters and their relationships, expressed as part of a *Context-based Parametric Relationship Model* (CPRM). The mathematical formulation for the CPRM model was initially defined in [8], as well as a basic description of the behaviour of the model. Here we extend the description of the behaviour of the model and focus on the requirements to develop the tool according to the model. This extension is necessary to detail the modelling of SQT, which was built based on what we consider to be the key requirements for analysing the Security and QoS tradeoff in future networks. SQT is considered as a handler for CPRM-based systems, and, in this paper, the steps for developing similar tools to SQT are provided[1]. Figure 1 shows the idea behind SQT. It is a knowledge-based tool that uses the information on the environment to provide information about the security and QoS parameters and relationships.

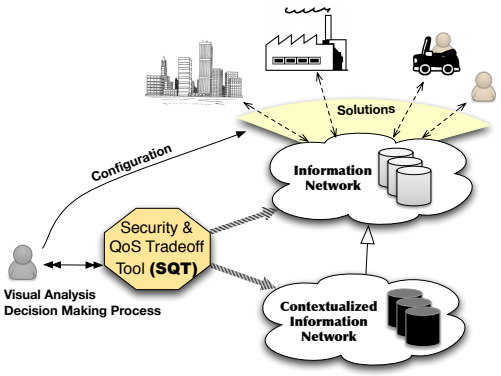


Figure 1: Security and QoS Tradeoff Tool.

The paper is structured as follows. Section 2 introduces the related work. Section 3 defines the CPRM model, defining the basic behaviour that should be implemented and the requirements for the integration of different contexts. Section 4 explains the steps that have to be taken to build SQT, based on the requirements imposed by the model. Section 5 provides the definition of the use case to be implemented in the analysis in Section 6. Our conclusions and future lines of research are given in Section 7.

2. Related Work

Analysing the current contributions it is possible to identify a set of trends in the analysis of security and QoS tradeoffs. Most of these contributions focus on service composition/selection, as

[1]In our case, we provide a prototype built in MATLAB that cannot be adapted to all the environments, but that we consider is very useful to understand the basic usability of SQT.

in [9], where model checking techniques are used to verify the composition of security services, or in [10], where a tool for evaluating the composition of security services based on *Multi-Objective Optimisation* (MOS) is provided. In [11] trust and QoS tradeoffs are analysed for web services composition. In this case, the selection of services concentrates on general services (not necessarily security services), and trust is a property of the services that must be considered during the agregation of services. In [12] the composition of security services is proposed for *Software Defined Networks* (SDN), where high-level approaches are feasible. Security is seen in some approaches like [13], as a requirement to protect the network against *Denial of Service* attacks for ensuring the QoS. In [14] an approach to provide security in SDN considering QoS guarantees is presented. Moreover, the security and QoS tradeoffs are also a problem in resourced-constrained environments that are connected to the Internet [11, 15]. In these environments defining the parameters, operations and the rest of the components and properties within a context, is key in identifying their relevance in the final composition of elements in the environment [16]. Alternatively, in [17] a model based on three static contexts (computing, physical and user) is defined based on a utility function in order to consider the user's preferences to capture fine-grane tradeoffs between security and QoS.

We conclude that most of the current approaches (i) focus on specific objectives (security and QoS tradeoffs using specific parameters or at specific layers, typically at the service layer), (ii) define generic models but do not consider partial-knowleadge of the environment (it is not always possible to predict the final mechanisms that will implement the properties), or (iii) do not consider the subjective perception of the user (what the user wants is not always best, but it is what they want).

We advocate assessing security and QoS tradeoff based on the analysis of parametric relationships, separating the parameters based on their type, different layers of abstraction and subjective values needed in the dynamic FI that affect the UC. These parametric relationships have to correctly define the dependencies between the security and QoS parameters. Our approach considers the composition of *things*, in that sense. We are not only interested in the services, but also in the low-layer characteristics or technologies that can be used to increase coexistence and cooperation in the network between the security and the QoS mechanisms.

3. Behaviour based on the Context-based Parametric Relationship Model

In order to combine the analysis of diverse mechanisms to provide security and QoS under a common framework, in [8] a *Context-based Parametric Relationship Model* (CPRM) is defined. The model defines the structure that a dependency-based system should have in order to provide useful information for analysis from a tradeoff perspective. It also defines the steps required to integrate new dependencies based on new conditions, to provide a new context. So, finally, the superposition of contexts, defines the new behaviour of the system, and this behaviour can be analysed based on a set of well-defined dependencies. In this section we extend the definition of CPRM to identify the basic requirements to build SQT.

3.1. Contexts & Subjectivity

The structure of a CPRM system is based on a set of parameters and the relationships between them, a set of operations (*op*) which define the effects on the dependent parameters, and a set of weights which define the relevant subjective and non-subjective components in the model.

For example, we can consider, subjectively, that *trust* is a key parameter for the system's survival at a specific time or in a particular context. With this assumption, trust would have a higher weight than the rest of the parameters in the dependencies system. When the purpose of the network changes, as a consequence of the security policies, or motivated by the environment of the user, then, the subjective values should also change. For example, when the environment is well known, as at home, the user could relax the relevance of the mechanisms of trust, because the knowledge of the devices surrounding them shifts their concerns towards other issues, like maybe the performance. In this sort of scenario, the user could consider the security as a consecuent of his/her location.

Moreover, the model considers non-subjective values, devised to define the impact of the cascade effect that a dependence can trigger. These values, are defined as weights linked to the dependencies in a *General Context* (GC), whose value is calculated, firstly, based on an approximation. Once the parameters have been *instantiated*, that is, at least one mechanism has been defined which implements the parameter, then, the weight for the inherited dependence is updated to the weight defined in

the *Particular Context* (PC). To reach this point, where different contexts are integrated or even interchanged, the definition of a clear set of structures is fundamental.

3.2. How the Model Schemes are Built

The integration of parameters is decomposed into different steps associated with what are called *model* schemes. So, a Parametric Relationship Model (PRM) defines a basic set of parameters (*Basic Context*, BC)[2], while GCs are present in *contextual* schemes as CPRM or $CPRM_i$, and, finally, PCs are present in *instantiated* models, denoted as $CPRM_i$. Considering that the scripts are built using sets of parameters and relationships, it can be said that: PRM $\subset$ CPRM $\subset$ $CPRM_i$, since the CPRM contains the parameters in the PC, but adds additional information (weights), and the CPRMi contains the parameters in the CPRM but adds additional information (e.g., new parameters). Specifically, a CPRM only has one GC, while a $CPRM_i$ has one GC, and, as minimum, one PC. Therefore, GCs and PCs are used as pieces, integrated into the model schemes, to build new model schemes. The steps for integrating contexts in order to have a $CPRM_i$ from a PRM, without going into great detail, are shown in Figure 2.

3.2.1. Non-contextual Scheme

A PRM is a non-contextual structure, which defines a set of parameters and their relationships. To improve the analysis of the influence degree between parameters, and the clasification of parameters, these are defined based on a type and a layer. Moreover, this separation makes the selection of a set of parameters based on a type or a layer possible. The model works with a set of predefined relationships in order to show the impact that the increasing/decreasing of a parameter has on the rest of the parameters. Note that, what we wish to do, is to detect what happens when a parameter is increased, decreased, enabled or disabled.

However, in a PRM, all the parameters have the same relevance. That is to say, in the relationship defined as $A \xrightarrow{+} B$ and $A \xrightarrow{+} C$, when A increases it has the same effect in B as C. This is impractical,

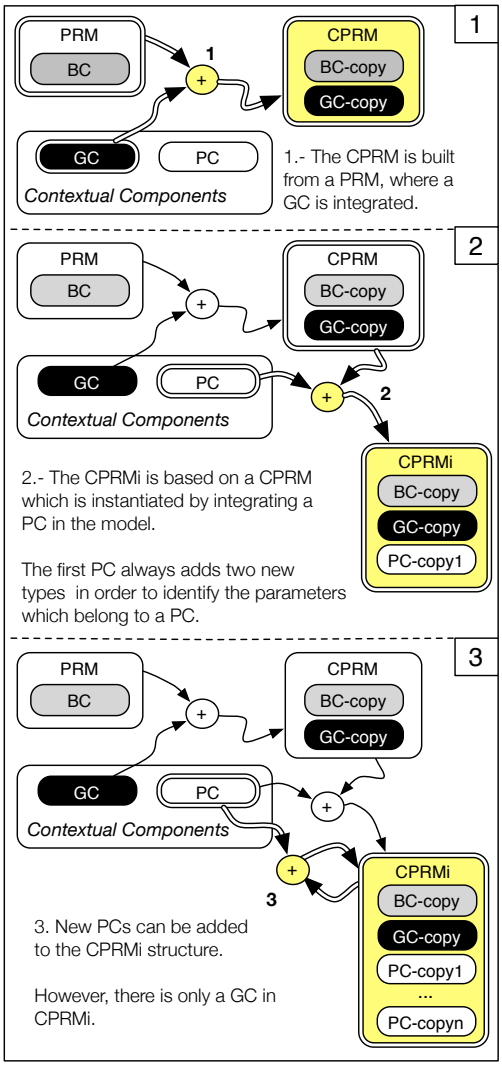


Figure 2: Steps for integrating contexts.

and also unrealistic, because there are some parameters that are more sensitive than others. For example, imagine a communication protocol, denoted as $C1$, that increases the energy consumption E and the computation time CPU, when used. We know that $C1$ is related to E ($C1 \rightarrow E$, $d(C1, E)$) and $C1 \rightarrow CPU$, however, in scenarios where E is a valuable resource, it may be preferable to have a protocol $C2$ that minimises E even at the expense of increasing the CPU.

Therefore, the CPRM-based models have been defined to take these differences into account.

[2]The BC is considered as an empty context with regard to GC and PC. So, in the following, the term *context schemes* refers to GC and PC, but not BC.

3.2.2. Contextual Schemes

A CPRM can be built based on an existing PRM, and a GC which defines the context, given as weights, for the elements in the PRM.

At this point, our schema knows that there are parameters which are more relevant than others and also that the relationships can have different values. However, it is not possible to improve the model in order to define the mechanisms that implement a parameter. For example, let us look at the previous relationship, which has been built on the basis of the general behaviour of the communication protocols. If we define a new communciation protocol $SecureC3$, with additional relationships, then, intuitively, $SecureC3$ can inherit the relationships of its generic predecesor, $C1$. Moreover, the scheme should learn that a communication protocol is secure, and then consider the relationships between the *father*, $C1$, and some of the consequents of the *child*, $SecureC3$, in order to improve the general behaviour of the model.

In order to permit this concept, where some new parameters are defined in order to provide a specific knowledge of the environment, we define the *instantiation* as the process of defining mechanisms, as new parameters, that implement parameters in the current CPRM. The new model or scheme is denoted as $CPRM_i$.

As Figure 2 shows, a $CPRM_i$ is built, based on a CPRM and a PC. Moreover, a new $CPRM_i$ is always possible from an existing $CPRM_i$ and an additional PC. In fact, PCs represent the dynamic nature of the model, as a result of a wider understanding of the environment. In other words, while the weights in the GC can be subjective, or even approximate, in a PC it is expected that the weights in the relationship will be accurate.

3.3. Rules & Action Rules

The previous steps for the integration of contexts are not possible without the definition of a set of rules to maintain the coherence of the definition of the model, through the process of instantiation of the CPRM based on the PC. In other words, the behaviour of the model can change according to *Action Rules* (AR), that are used when a rule (R) is not satisfied, thereby making the model consistent again. ARs are defined according to the rules shown in Table 1, which express the correct behaviour expected in the model. In order to clarify the application of ARs, we use the example shown in Figure 3, where the final relationships in a $CPRM_i$, in which two PCs are integrated (using ARs), are illustrated.

3.3.1. R1: Basic set of parameters

Rule R1 is set to ensure that the PRM contains a basic set of parameters, so that diverse relationships in heterogeneous environments can be defined. Therefore, if a new PC contains parameters without a parent in a PRM, it can be interpreted in two ways: the parameters in the new PC have been defined badly, or the parameter without parents defined is a new parameter to be considered as part of the PRM basic set (as part of the BC). We have to assume that the PC is well defined, so, when a new parameter is identified, it is added as part of the BC in the PRM (AR1), making it a new parent to be considered. If not, the definition of BC would be unstable, because in this case the PRM does not provide all the possible parameters that can be instantiated. Then, in accordance with AR1, in order to integrate PC2, the parameter $P11$ is added as a new PRM parameter. However, usually such additions occur at an early stage of recognition of the environment, or as part of the aggregation of a GC (e.g. $P6$, $P7$).

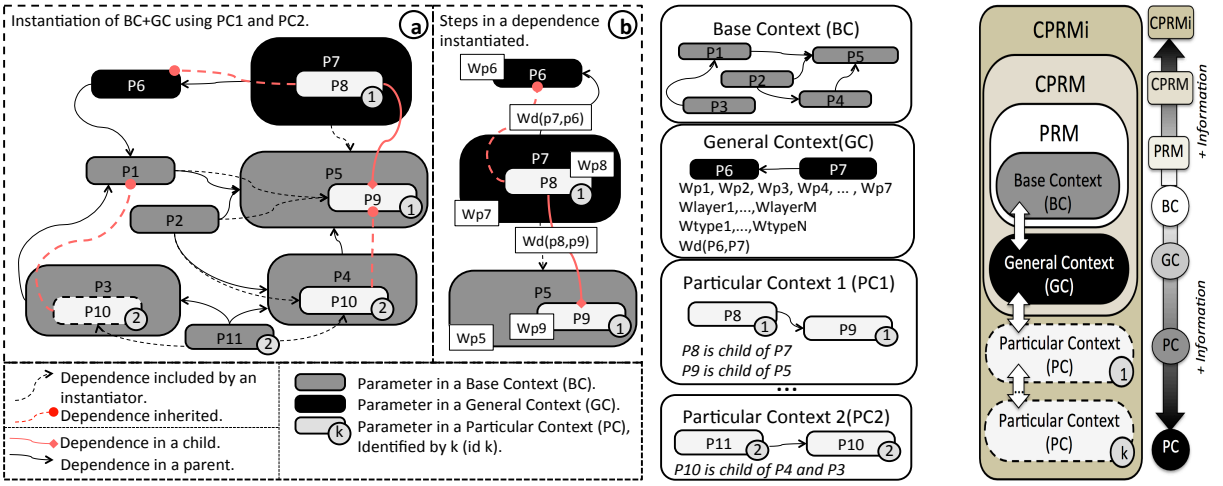
3.3.2. R2: Inheritance from the Instances (new behaviour)

On the other hand, rule R2 protects the inheritance of the parents' relationships. As the parents are considered as the most general parameters, it is expected that these parameters define all the possible relationships. Moreover, as the instances inherit their relationships from their parents, then, the relationship between two parameters is not possible if their parents are not related. For this reason, the relationship between two parameters whose parents are not related, makes the model inconsistent, and is avoided by adding new relationships between the parents (AR2).

However, new relationships added between parents to make the model consistent according to rule R2, cannot have the same weight as the relationships between their instances. This is because any new parameter that instances the parent, inherits this new relationship, and the default weight. So the weight for this new relationship between parents, set by AR2, is 0 ($w_d(k, z) = 0$). This is the case of the new relationship added between $P7$ and $P5$.

Table 1: Rules (R) and Action Rules (AR).

	Rule		Action Rule
R1	A parameter in the PC is related to at least one parameter in the PRM (parents).	AR1	If not, the independent parameter is considered as one new parent and added to the PRM to make it consistent.
R2	Given $P(x)$ and $P(y)$ the list of parents of x and y, respectively such that $P(x) \cup P(y) \subset PRM$. If $d(x, y)$, exists, then $\exists k \in P(x) \wedge \exists z \in P(y)$ such that $d(k, z)$.	AR2	Otherwise, said relationship between parents has to be added to the PRM in order to make it consistent, with $w_d(k, z) = 0$.
R3	A parameter in the PC inherits the relationships of its parents, by default: if z belongs to $P(x)$ and $d(z, k)$, then $d(x, k)$ is possible, with weight $w_d(z, k)$ by default.	AR3	The relationship $d(x, k)$ is added with $w_d(z, k) \forall k$. If $\exists p$ such that $k \in P(p)$ and therefore, according to R2, $d(x, p)$, $w_d(x, p)$ don't change.
R4	A parameter x inherits the layer of its parents and the type of its parents. When $\exists k, z \in P(x)$ such that $type(k) \neq type(z)$, then $type(x) =$ list of $\{type(k), type(z)\}$.	AR4	The decision model can fix the type of the layer of a parameter, but, even so, the final layer and type match with the layer and type of a parameter p in $P(x)$.



(a) Final CPRM instance ($CPRM_i$)

(b) Models and Contexts

Figure 3: Contexts as Integration Components

3.3.3. R3: Inheritance from Parents (known behaviour)

Note that, the previous relationship, $d(P7, P5)$ is different from $d(P4, P5)$ which was defined in the PRM. Therefore, R3 defines the inheritance of relationships from the parents to the instances. For example, $P9$ and $P10$ are defined in PC1 and PC2, respectively. Therefore, as $P4$ depends on $P5$, so $P10$, defined as a child of $P4$, inherits this relationship with the weight defined by default (AR3), $w_d(P4, P5)$. So, this relationship implies that $P10$ will be related to $P9$ as a new possible combination.

Moreover, when the parameter $P10$ is added, it inherits the relationships of its parents based on AR3. It means that $d(P10, P1)$, $d(P10, P9)$ are possible, with $w_d(P10, P1) = w_d(P3, P1)$ and $w_d(P10, P9) = w_d(P4, P5)$, respectively. Furthermore, the relationship $d(P2, P10)$ is added as a consequence of the relationship $d(P2, P4)$ defined in the PRM. Then, according to AR2, $d(P2, P3)$ has to be added too, because $P10$ is also a child of $P3$, and $P2$ is related to it. However, $w_d(P2, P3) = 0$, and this does not affect $P10$, which is only affected by the inheritance of the relationship $d(P2, P4)$.

3.3.4. R4: Analytic behaviour

Finally, R4 determines that all the parameters inherit the type and layer of their parents, and AR4 forces this. So, it is unnecessary to provide this type of information in the PC. However, note that, when AR1 is applied, the creation of the new parameter may need this additional information. In this case, the type of the new parameter is *instantiated*, because it was built as part of a PC. For this reason, it is very important that new parameters are added in an early step (in a BC or GC), because, although

the model considers these types of events, adding new parents from a PC is not the objective and breaks the initial purpose of maintaining a stable context separate from the dynamic context.

3.4. Requirements for the Integration in a $CPRM_i$

The action rules shown in Table 1 are taken into account when a PC is used to instantiate a CPRM. However, before setting up the rules, a new structure is needed in order to represent an instance of the CPRM. The $CPRM_i$ has to be defined taking into consideration a set of requirements: (i) independence from the original CPRM, (ii) coherence between parameters in the model (Table 1), (iii) adaptation capability, this is, acceptance of a new PC to be added to the existing one, and (iv) ability to return to the original CPRM behaviour.

Although CPRM is an extension of PRM that considers context-based behaviour, $CPRM_i$ has to be considered as an *instantiation of a CPRM based on a PC*. The $CPRM_i$ is not a new version of PRM or CPRM, because its purpose is not to define a model or implement functions. Rather, its purpose is to change its behaviour based on different contexts. Thus, a $CPRM_i$ is always built based on a CPRM and a PC, but when it is created, the original data in the CPRM should be cloned in the $CPRM_i$ to make it independent.

A $CPRM_i$ represents the final behaviour of the system, in which all the mechanisms and technologies that are relevant are finally chosen by the administrator and taken into account when extracting relevant information of the model. The current definition of $CPRM_i$ has been built, based on the following steps:

1. The $CPRM_i$ adds the special types *Instance* and *Instantiated* to the model.
2. The type Instance will be the type in the model for the parameters included from the PC. During the inference process the given parameters take the original type of their parents according to a set of rules. With these tags in the model it is possible to properly identify which parameters belong to a given PC. As a result, it is possible to return to the original CPRM behaviour.
3. If a parameter y is Instantiated, that is, if $\exists x \in PC$ such that $y \in P(x)$, then in the $CPRM_i$ the parameter becomes a new layer. The new layers which represent instantiated parameters are separated from the rest of the layers, defined in the model. When the PC is retrieved, the parameters which cease to be instantiated return to the original list of parameters in the model.
4. When the $CPRM_i$ receives a new PC, the new parameters are integrated in order to maintain the coherence in the model, based on the rules in Table 1.

When an instantiated parameter is represented as a layer, this adds the possibility of calculating the effect that the whole layer has on the performance of the system, thereby making it possible to evaluate the tradeoff between different mechanisms.

4. Building the Security and QoS Tradeoff (SQT) Tool

In this section, the steps to be taken for the integration of the components of the model, being implemented in a prototype built in Matlab, are detailed. A component-based architecture is defined in order to enable the steps for the integration, to satisfy the previous requirements of independence between models, coherence and adaptation capability. The user sets the contexts using the GUI shown in Figure 5.

4.1. Components-based Model

The design of SQT is based on interchangeable components, following the behaviour described in Section 3. Also, in order to facilitate the analysis (make comparisons and so on), it is necessary to be able to return to the previous version of the model, for example, by removing the last context added, or even by building new contexts and by removing any of the integrated contexts (not necessarily the last one added). Moreover, these changes should not affect the definition of the model itself; the model has to be consistent. So, following the rules, and the previously defined ARs, the multiple integration of interchangeable contexts, consistently, is guaranteed. In addition, the integration chain and the data structures for allowing said properties also need to be defined, following the set of criteria that are described in this paper.

4.1.1. Generating Self-Defined Contexts

To provide the functionality of interchangeability, taking contexts as components, the integration of contexts is performed in SQT based on the diagram shown in Figure 4. Specifically, the activity

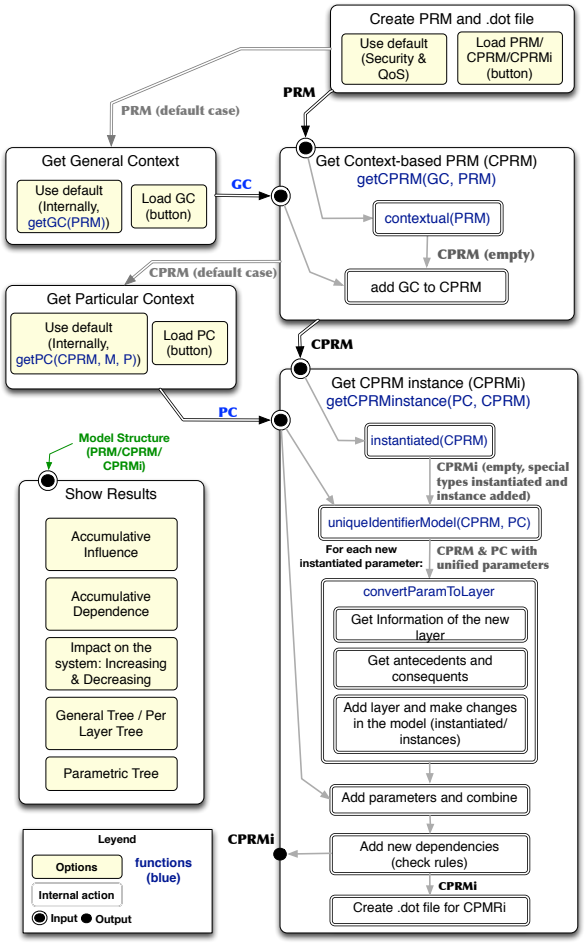


Figure 4: Instantiation process in SQT.

diagram shows the creation of a $CPRM_i$ from a PRM. To do this, at each step, the intermediate structure necessary to complete the model definition is built. Note that this procedure is defined in such a way, as to build a default model, used as the example to be modified. In general, the GC or PC are previously defined, not randomly generated.

Moreover, as our goal is that any context can be extracted from an existing model, the functions used to generate contexts to be used as examples, also have to be able to extract contexts when the input received is adequate. This is the case for the functions getGC and getPC, that are used to extract (if it exists) or generate (if it doesn't exist) a context based on a given parametric model structure. This behaviour depends on the input given to these functions. In the case of getGC, when the input is a PRM, that is, a model without a preassigned context, then, the function generates a default GC for the parameters in the PRM [3]. Otherwise, if the input is a CPRM or a $CPRM_i$, let us say, any structure with a GC preassigned, then, the GC of the structure will be returned.

Similarly, getPC only returns the PCs assigned to a structure when they are defined, a condition that only a $CPRM_i$ may satisfy. So, when getPC receives any other model structure, it returns a new, randomly generated PC, in accordance with the parameters given in the model. The parameters of the type *instance* will be fictitious, only for testing purposes[4].

4.1.2. Assigning the Contexts

Then, the next step is the assignation of contexts to models, which is done by the functions getCPRM and getCPRMinstance. First, getCPRM generates a new CPRM, based on the GC and PRM/CPRM provided. In order to assign the GC to the model, the latter is previously converted to a contextual-based model, if it is not one already. Second, the function getCPRMinstance generates a new $CPRM_i$, given a PC and a PRM, a CPRM, or a $CPRM_i$. In this case, when the model given as input is not an instantiated model (not a $CPRM_i$), then, the model is converted to a contextual-based model. After that, it is converted to make it compatible with an instantiated model. Finally, the PC is integrated in the model. However, if getCPRMinstance gets a $CPRM_i$ as input, the process followed is more complex, since various PCs can coexist in the same $CPRM_i$. So, although the main objective is the same, to obtain a new structure given a context and a model, getCPRMinstance requires a much more detailed analysis than getCPRM.

Note that it is expected that an instance of a CPRM, that is, a $CPRM_i$, will be more dynamic than a CPRM. It is also motivated because, if the system is well defined, the PCs should be interchanged more frequently than a GC, that, although it can be modified, it is assumed that it will be the most stable definition of the context. Moreover, when the GC is retrieved, the PCs (if they exist) are also removed from the model, because they were set up based on the existence of a GC.

[3] After that, the GC can be applied to the PRM, generating a new CPRM, as Figure 2 shows.

[4] M and P indicate the number of parameters of type *instance* that will be generated for each parameter of the model given as input.

4.1.3. Adaptation to a Dynamic Behaiour

It can be considered that the integration of new PCs into a $CPRM_i$ is the most delicate step. For example, the new PC defines parameters that, being different from those in the $CPRM_i$, could be taken to be the same set of parameters and this has to be taken into account. This scenario could occur, given that the same parameter can be part of several PCs. For this reason, defining the parameters based on a set of identificators (instead of only one) is very important, in order to perform the mapping between parameters during the integration step, thus avoiding the overlapping between different types of parameters that could be taken to be the same one. This is only one example of the issues that have to be considered when integrating a PC into a $CPRM_i$. These issues are managed by SQT, using the function getCPRMinstance.

This function, given a CPRM, builds the skeleton of a $CPRM_i$ structure (using *instantiated*), adding the new types and additional fields (Table 3) required to satisfy the properties of the model. After that, it completes the model by adding the information in the CPRM, and, finally, it integrates the new PC into the model. If the function receives a $CPRM_i$ as input, then, it is only necessary to perform the final step of integrating the new PC in the $CPRM_i$. Moreover, this last step is the most complicated, because it is responsible for avoiding the overlapping of parameters and the additional issues related to the integration of parameters. If this step is not properly achieved, then, probably, the context will not be retrieved once it has been integrated, or, even worse, the final model could be incoherent.

4.1.4. Backup-able Integration

As Figure 4 shows, the steps after the function *instantiated* are critical. After that point, the unification of parameters, that is, the mapping, is done, taking into account the definition of the parameters given in the model. Then, once the model has been unified, getCPRMinstance identifies the parameters that will be instantiated, and generates new layers for them. This means selecting those parameters p (instantiated), so that there are parameters defined in the PC which instantiates p (instances), let us say p such that $\exists p2 \in$ PC, $p \in P(p2)$. In order to do that, the function convertParamToLayer is used, to return the definition of the new layer. These layers store all the information needed so that in the case that the PC is removed, the model will be able to return to its previous behaviour, prior to the instantiation using the PC.

The last step in the integration is to determine the relationships that will be inherited by the instance of a parameter, and, moreover, generate the new dependencies required to maintain the coherence in the model. This step is done based on Table 1, as defined in Section 3. After that, the $CPRM_i$ is complete, and can be exported to a *.dot* file, that is interpreted as a graph using GraphViz.

Finally, regarding the analysis, all tests that are possible to do on a PRM are possible on a CPRM or a $CPRM_i$. The difference is, that while a PRM is static, a CPRM also presents a subjective view of the context of the network, giving more relevance to parameters, relationships or transactions in accordance with the management priorities or deep knowledge of the network.

In addition, a $CPRM_i$ enables the integration of dynamic contexts, based on the specific information of the current state of the network. These final contexts, the PCs, are not only more variable and fleeting contexts than the GCs, they are also more specific.

4.2. Data Model Structures

As we can deduce from the previous sections, any structure PRM, CPRM, $CPRM_i$, PC or GC, has a predefined format in which they are created and used. SQT holds all these structures as part of a general structure, S (Exp. 1, Table 2), that can be saved, as workspace.

Table 2: Model & Context Structures

$S = \{D1, D2, D3, D4, D5\};$	(1)
$D1 = \#prm, nxtID, \{\{PRM_{id}, id, info, file\}, ...\};$	(2)
$D2 = \#cprm, nxtID, \{\{CPRM_{id}, id, info, gcid, file\}, ...\};$	(3)
$D3 = \#cprmi, nxtID,$ $\{\{CPRMi_{id}, id, info, gcid, pclist, file\}, ...\};$	(4)
$pclist = [PC_{pcid}, PC_{id2}, ...];$	(5)
$D4 = \#gc, nxtID, \{\{GC_{gcid}, gcid, info, file\}, ...\};$	(6)
$D5 = \#pc, nxtID, \{\{PC_{gcid}, pcid, info, file\}, ...\};$	(7)
$GC(1, 1:2) = \{NL\{id\_layer1\ w_{1,1};\ id\_layer2\ w_{1,2}; ...\}\};$	(8)
$GC(2, 1:2) = \{NT\{id\_type1\ w_{2,1};\ id\_type2\ w_{2,2}; ...\}\};$	(9)
$GC(3, 1:2) = \{NO\{id\_op1\ w_{3,1};\ id\_op2\ w_{3,2}; ...\}\};$	(10)
$GC(4) = \{\};$	(11)
$GC(5, 1:2) = \{NP, NProp\};$	(12)
$GC(6:(5+NP), 1:NProp) = \{id\_param1\ w_{6,1}; ...\};$	(13)
$GC(6+NP, 1:2) = \{ND, \{id\_dep1\ w_{6+NP,1}; ...\}\};$	(14)
$PC(1, 1:4) = \{NP, Nprop, ND, \{IDpc, descrip.\}\};$	(15)
$PC(2:(1+NP), 1:NProp) =$ $\{id\_Parents, id, name, w_{p,1}\};$	(16)
$PC\{3+NP\} = \{id\_ParamA, idOp, id\_ParamB, w_{d,1}; ...\};$	(17)

Table 3: Fields for Data Structures in Model Schemes

Row,Column:Purpose	PRM definition	CPRM (changes in PRM)	$CPRM_i$ (changes in CPRM)
1,1-2: Info. layers	Number of layers (NL) + Properties of layers	Adds the property *weight of the layer* w_l.	Defines special layers for the parameters instantiated (parents).
2,1-2: Info. types	Number of types (NT) + Properties of types	Adds the property *weight of the type* w_t.	Adds two special types: *instance* and *instantiated*
3,1-2: Info. Ops.	Number of operations (NO) + Properties of operations	Adds the property *weight of the operation* w_o.	-
4,1-2: More Info.	Directory by Default (DD)	-	Adds information on the number of PCs integrated.
5,1: NP	Depends on the model.		
5,2: NProp	5	6	6
5:(5+NP),1-NProp: Parameters	Properties of parameters	Adds the property *weight of the parameter* w_p.	The parameters instantiated changes its layer by the new one created as a consequence of the instantiation.
6+NP,1: Dependencies	Dependencies before being processed (BD) or Matrix of dependencies processed (MD)		
6+NP,2-3: Dependencies Processed	Matrix of zeros NPxNP + BD	Matrix of weights NPxNP + BD	-

So, from the implementation point of view, it is possible to assume that S represents the data model, composed by the model structures and the context structures, where: PRM_{id}, $CPRM_{id}$ and $CPMRi_{id}$ are model schemes (cells in MATLAB, similar to Figure 6) with an identifier assigned (id), $info$ shows information relevant information to the model, and $file$ information about the location of the model scheme. The value $nxtID$ allows knowing the next identifier to be assigned in the case that a new model scheme is added. The description of the context schemes is quite similar, after the value $nxtID$, the cell with the information about the scheme starts with the context scheme (cell in MATLAB with the description of the context). Moreover, at the begining of each section Dk, $k = 1 : 5$, the number (#) of scripts is indicated.

In Table 3 the physical differences between the three types of model schemes are shown. Based on these discordances, SQT can identify when a model structure is a PRM, a CPRM or a $CPRM_i$, and then, it manages the operations defined according to the type of structure and its definition. Moreover, the commonalities between the models, makes the component-based integration feasible.

Note that, any element in a PRM has to be identified by two identifiers, minimum: the unique identifier value (numerical value, id), and the name of the element (string). Both identifiers are present in the properties of the elements, that is, in the properties of layers, types, operations and parameters. The properties of the elements also identify the visual representation of the elements in the Matlab diagrams, or in GraphViz (color and shape).

Moreover, once the particular dependencies of a parameter with the rest have been calculated, this matrix, specific to the parameter, denoted as *Parametric Map* (PM), is stored as a property of the parameter. This matrix is recursively generated, and defines all the relationships where the parameter is involved (that is, a NPxNP matrix). The drawback is that these PMs (one per parameter) may require a large amount of space in the data structure; which is the price to be paid for not having to re-calulate the PMs more than once. The PMs, are calculated based on the *Parametric Table* (PT), that is a matrix defined in [2][5], where all the direct relationships between parameters are shown. So, if $\exists a, b \in PRM$ such that $d(a, b)$, then, $PT(a, b) > 0$. Initially, this knowledge is expressed through *Brute Dependencies* (BD), $A \xrightarrow{op} B$, using the numeric identifiers *in a cell at the end of the definition of the model scheme*. So, for example, $A \xrightarrow{op} B$ is expressed as id_A, id_op, id_B, and, in a contextual-based model, the weight of the dependency is included at the end of the relationship: $id\_A, id\_op, id\_B, w_{d(A,B)}$.

Finally, it is important to remark, that the PRM provides the basis for building the CPRM models and the $CPRM_i$ instances (or instantiated models). However, there are differences in the data structure which cause huge changes in the calculation process achieved by SQT. So, while the CPRM structure represents a turning point between a PRM and an instantiated model, the really significant changes are found in the definition of the $CPRM_i$ structure. This is principally due to two key factors: the definition of the special types *instance* and *instantiated*, and the conversion of instantiated parameters

[5]The parametric table is a table NPxNP that contain values different from 0 in the position (x,y) when the parameter x in a row is related with the parameter y in a column.

to layers. These factors, coupled with the ability of the model to modify or restore itself, by means of the elimination and the aggregation of contexts, are a big change from the non-instantiated models, which are relegated to a more static function.

4.3. Context Structures

The definition of specific data structures for GCs and PCs is necessary in order to manage them as interchangeable components. Two skeletons for GC and PC are provided using Exp. 8-14 and Exp.15-17, respectively (Table 2).

Some fields in the context structures, GC and PC, are the same as those defined in the model schemes. This is necessary, because the context schemes will be part of the contextual model schemes, and, therefore, a common part between them will be required to match them, like pieces of a puzzle. Indeed, the context schemes are intended to change the properties in the elements of the model (parameters, relationships, types ...), so, they have to at least be able to identify these elements and the new values.

While NL, NT, NO, and ND are defined in Table 3, in the GC and the PC, the fields $NProp$ and NP refer to the context structure itself. A context structure defines its own extension. For example, in the current version, in a PC, the number of properties for a parameter, is equal to five ($NProp = 5$): the list of identificators of the parents of the parameter ($id\_Parents$), the identificator of the parameter (that can be modified if the ARs are applied), the name of the parameter, and the weight (w).

The rest of the symbols indicate the identifier of a component in a PRM (e.g., layers and types) and a weight (w)[6]. Therefore, a GC is for adding weights to an initial PRM, or to change the weights in a CPRM.

Given a PC, when it is integrated in a CPRM, the parameter $p \in PC$, will be of type *instance*, and its type and layer are inherited from its parents, which take the type *instantiated*. In both cases, these modifications are marked with the identifier of the PC ($IDpc$ in Exp. 15) which introduces the changes.

4.4. Graphical User Interface

The first prototype of SQT provides a *Graphical User Interface* (GUI), designed to show all the options available to the user in a single window, divided into seven sections or panels (Figure 5):

1. PRM Panel. Load a PRM from an ".m" file, or load a default PRM. It is possible to load a CPRM or a $CPRM_i$.
2. Panel for operating with GCs. Intended for selecting the GCs and model structures that allow a GC to be added or extracted.
3. Panel for operating with PCs. Intended for selecting the PCs and model structures that allow adding the PCs, or extracting them.
4. Working panel. This panel can be used once the first PRM has been loaded. Includes the operations defined by the model.
5. General Panel. Intended to act on any of the elements defined in the workspace (S). Through this panel, it is possible to close or delete any element in S.
6. State Panel. Intended to save the workspace (all the context and model structures in S), as well as to load an entire workspace, that has been previously saved in a ".m" file.

In addition, the GUI includes one informative panel that shows information on the operations performed using the GUI, and any errors that may have occured.

It is important to note that any model structure can be saved into a new file with extension ".m". So, the user can modify an existing example (e.g. default sample) so that it builds its own models and contexts. The functionality for saving the default models and contexts, is designed to simplify the learning process of the user so they can use SQT easily.

5. Use Case: Authentication in WSN

The use case, to be implemented in Section 6, focuses on a scenario where a WSN may be deployed, and two authentication mechanisms are available, CAS and DAS. To consider sensors as possible devices involved in the network, it is necessary to take into account parameters such as energy or power consumption, amongst others. The following sections separate the parameters in the base context, considered to implement the behaviour of a WSN, from the parameters in the particular context, defined to implement the use case related to the authentication in WSN.

[6]We target these weights with two identifiers to indicate that these are different.

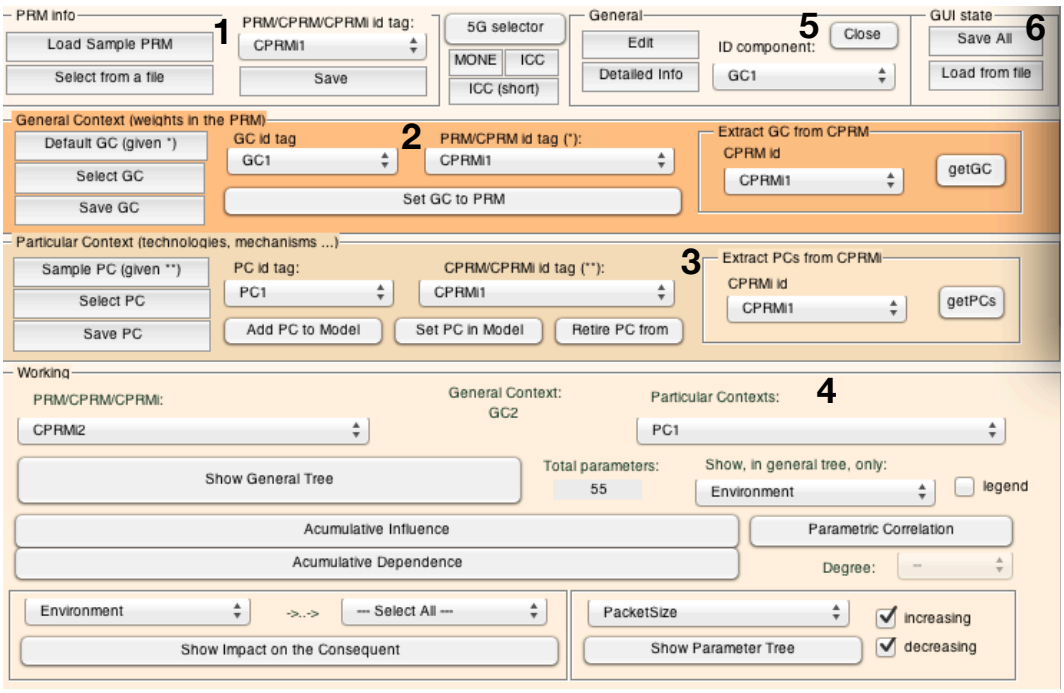


Figure 5: SQT: Graphical user interface.

5.1. Parameters in a Base Context

The parameters in the base context, shown in Table 4, were taken from [2], as part of a detailed study, where the convergence of WSN with other networks in the *Internet of Things* (IoT) was analysed. Here, the basic parameter set is reduced so as to be able to work with it from a theoretical point of view [7].

Although the default GC, given these parameters, is set to 1 for all the parameters ($\forall p$ such that the parameter $p \in PRM, w_p == 1$), it is possible to set a subjective GC, based on our own priorities. For example, increasing the relevance/impact of the parameter *Encryption*, will cause any parameter related to Encryption in the antecedent, that is, $\forall y$ such that $Encryption \rightarrow y$, to be more affected than the rest. The parameters affected by the increase/decrease of the parameter Encryption, can be consulted using the parametric tree. However, the final impact on the parameters may vary according to the type of the relationship defined between the parameters (*op*, see [8]), and the weights assigned to them. From now on, all the weights for the parameters (the relevance), are set to 1. Moreover, the relationships are defined in the default GC with value 1 for all the relation-

Table 4: Parameters for a Base Context

	HIGH-LEVEL REQUIREMENTS
QoS	Reliability, Fault Tolerance, Availability
Security	Authentication, Authorisation, Confidentiality, Integrity, Trust, Privacy
	LOCAL PROPERTIES
Resources	Power Consumption, Memory, Rayleigh Channel, Energy, ComputationTime
Security	Anti-Tampering, Encryption, Public Key Cryptography, Certificate, Symmetric Cryptography, Secure Key Exchange, Secure Key redistribution, Key Generation, Signature Scheme
	COMMUNICATION
QoS	Data Rate, Packet Size, Signal Strength, Data Transmission, Transmission Time, Transmission Power
Characteristics	Time-sleeping, Required-time-on, Routing Protocol
Consequence	Retransmmission
	MEASUREMENTS
QoS	Throughput, Delay, Jitter, Packet Loss, Response Time, Bit Error Rate (BER)
	ENVIRONMENT
QoS	Allowable Bandwidth, Error Probability
Attacks	DoS, Malicious Devices
Consequence	Interference, Congestion, Overhead, Fading, Shadowing, Noise

[7] The relationships considered between the parameters, as well as any other related information about the significance of these and other parameters can be consulted in [2].

ships ($w_d = 1, \forall d : A \rightarrow B$ such that $d \in PRM$). These values can be modified in the GC, but, in our case, the changes will be made using an example of instantiation of parameters. In other words, we use PCs to assign final weights w_d to the dependencies in the model[8].

5.2. Parameters in a Particular Context

In the following sections, the PC shown in Table 5, will be integrated in the CPRM which contains the parameters shown in Table 4. The weights in Table 5 have been fixed according to the work done in [18], while the information about the specific relationships defined in Table 5 is extracted directly from the same, aforementioned paper. The rest of the relationships, defined in the PRM, are taken from our previous paper [2].

Table 5: Weights w_d according to [18]

	Dependence			
Instantiated	Antecedent	R	Consequent	w_d
Authentication	CAS	+	ECDSA	1
	DAS	+	ECDSA	1
	CAS	$\neg c$	Memory	0
	DAS	$\neg c$	Memory	5
	CAS	c	PacketSize	5
	DAS	c	PacketSize	1
	CAS	c	Certificate	2
Signature Scheme	ECDSA	$\neg c$	Energy	1
	PairingBased	$\neg c$	Energy	5
	ECDSA	c	Computation Time	1
	PairingBased	c	ComputationTime	5

So, the parameters to be instantiated are, in this PC, *Authentication* and *SignatureScheme*, and the instance parameters are CAS, DAS, ECDSA and PairingBased. Note that both *parents*, Authentication and SignatureScheme, are defined in the BC in Table 4. Therefore, they have their own relationships in the PRM. The definition for CAS and DAS can be found in [19]. The *Certificate-based Authentication Scheme* (CAS) uses the user's public/private key pair to provide authentication. This requires the use of certificates. Instead, the *Direct Storage based Authentication Scheme* (DAS), avoids the use of certificates to reduce the overhead. To do this, DAS stores the current user's ID information and their public keys. Both schemes use the *Elliptic Curve Digital Signature Algorithm* (ECDSA) to sign the broadcast messages.

So, the focus here is to integrate the specific information or context behaviour, represented in Table 5, inside our predefined model. Once this new information has been integrated, the final instantiated model, will provide specific information about the authentication and signature schemes allowable in the final environment.

The tradeoff is in the effect that these mechanisms can have on the rest of the parameters already defined in the model, that were not previously considered in the specific papers or any other source from where the PC was extracted. It is important to remember that, with each new integration in the model, the ARs defined in Table 1, can define new dependencies. So, when the parameters increase, the $CPRM_i$ can show a new behaviour, completely different to the original CPRM, which is normal, because the $CPRM_i$ is based on the context.

6. Assessing Security & QoS tradeoff

Here the usability of the tool will be fully tested. All the figures shown have been generated by SQT.

6.1. Setting up the model

In this first step we only load the model from a file, and show the influence and dependence degrees. Figure 6 shows a part of the file *.m* in which the PRM, which contains the parameters shown in Table 4, was defined. It is important to note that the parameters in the PRM are parents, that is, non-instantiated parameters. Even so, from this kind of scheme, we can extract some valuable general information shown in the next subsection.

Note that SQT allows the generation of a PRM by default, that can be saved (in a new file *.m* similar to that shown in Figure 6) and modified as we wish. The same can be done for any other structure or component used by SQT, even the workspace can be saved and loaded again.

Once the model has been loaded, the working panel can be used to show the accumulative influence and dependence degree [8]. This is a general view of the impact that the parameters have on the model (Figure 7). For example, as can be observed, in the PRM chosen, the parameter Trust does not affect the rest, while in other scenarios, such as mobile platforms, this parameter is directly

[8]Given Table 1, changing the value of the relationships where a parameter *parent* is involved is of very little use when, in the next step, the instances of the parameter will define the same relationships using its own weights.

```
PRM={
 %LAYERS
 [  5 ] {  [  1 ] 'HighLayerRequirements' 'red';
           [  2 ] 'LocalProperties' 'orange';
           [  3 ] 'Communication' 'blue';
           [  4 ] 'Measurements' 'green';
           [  5 ] 'Environment' 'pink' } [  ] [  ] [  ];...
 %TYPES
 [  6 ] {  [  1 ] 'Class' 'coral' 'egg' [  1 1 0 ];
           [  2 ] 'Performance' 'coral' 'trapezium' [  1 0 1 ];
           [  3 ] 'Characteristics' 'cadetblue2' 'oval' [  0 1 1 ];
           [  4 ] 'Security' 'brown1' 'box' [  1 0 0 ];
           [  5 ] 'Attacks' 'brown3' 'septagon' [  0 0 1 ];
           [  6 ] 'Consequences' 'burlywood1' 'hexagon' [  5.000000e-01 5.
 %OPERATIONS
 [  9 ] {  [  1 ] '+' [  3.000000e-01 ] [  1 ] [  0 ];
           [  2 ] '-' [  6.000000e-01 ] [  -1 ] [  0 ];
           [  3 ] 't' [  9.000000e-01 ] [  1 ] [  -1 ];
           [  4 ] 'n+' [  1.200000e+00 ] [  0 ] [  -1 ];
           [  5 ] 'n-' [  1.500000e+00 ] [  0 ] [  1 ];
           [  6 ] 'i+' [  1.800000e+00 ] [  1 ] [  1 ];
           [  7 ] 'i-' [  2.100000e+00 ] [  -1 ] [  -1 ];
           [  8 ] 'c' [  2.400000e+00 ] [  1 ] [  -1 ];
           [  9 ] 'nc' [  3 ] [  -1 ] [  1 ] } [  ] [  ] [  ];...
 'imagesFolder' 'ShortDescription' [  ] [  ] [  ];...
 %PARAMETERS & PROPERTIES
 [  48 ] [  5 ] [  ] [  ] [  ];...
 [  1 ] [  2 ] [  1 ] 'Reliability' [  ];...
 [  1 ] [  2 ] [  2 ] 'Availability' [  ];...
 [  1 ] [  2 ] [  3 ] 'FaultTolerant' [  ];...
```

Figure 6: Part of the content of a PRM (file *.m*).

related to the user's experience. Here the model has been simplified, considering only a part of all the relationships that can be found in the general model provided in [2]. To the contrary, as can be observed, the influence of the parameter Authentication has been especially considered, and these relationships are provided with special interest.

6.2. Analysis of parameters, prior to the instantiation

Here we show the parameter tree[9] for those parameters that have not yet been instantiated, but will be instantiated in the next step. The following analysis focuses on, in particular, the parameter Authentication, whose parametric tree is shown in Figure 8, given, according to the definition of the PRM defined in Section 5. Specifically, the figure illustrates the parameters that are affected when the Authentication is provided (aka *increased*, that is Δ). This dependency tree is quite different from the parametric tree calculated for decreasing the Authentication (∇), shown in Figure 9 (case 1).

Figure 8 can be interpreted as follows: when Authentication is provided, based on the current literature, the parameters ResponseTime, PacketSize, Memory and SignatureScheme can all be affected by an increase, or, in other words, if they are properties, they should be provided. Moreover, each one of these parameters have their own dependencies that were originally defined in the PRM. So, for example, it is known that an increase in the memory can create an overhead in the system, if there are more services which need this memory in order to work properly. In that case, the interference is considered as the probability that the system will collapse, decreasing the performance. Intuitively, under these conditions, the probability of failures in the system increases, as does the delay. Both, delay and error probability may cause packet loss, even when they are considered at the local layer, because these finally affect the capacity of the system to respond to the neighbouring demand. Once packet loss has been affected, the probability of retransmissions increases, and, in a WSN, this means that the nodes cannot enter into a sleep state to save energy as much as they should. Therefore, in a WSN, if the time for the antennas to be in power on increases, it implies less time in inactive mode, so, the power consumption increases, which results in a decrease in global energy.

When, in a parametric tree, there are leaf nodes, it can mean two things: first, there are no relationships defined for this parameter. Second, when this parameter is affected by the corresponding effect, increasing (Δ), or decreasing (∇), this effect has no result; there is nothing that can be done.

For example, note the case of $Authentication \xrightarrow{+} PacketSize$. As a positive relationship ($+$) was defined, then, PacketSize is only affected if Authentication increases. Otherwise, the system has no information, and in that case, the influence chain stops in PacketSize. This difference can be observed in Figure 8, where an increase in Authentication triggers an increase in PacketSize, while in Figure 9, a decrease in Authentication has no effect on PacketSize, so, in the latter case, there is no propagation through this branch.

The *complete* relationships (c and $\neg c$) are defined in both cases, for increasing and decreasing, and therefore can be observed in both trees [10].

In the following sections, in order to broadly outline the use case, the parameter trees for decreasing will be considered. This is because the dependency trees, although simplified for testing, are highly extense, and the trees for decreasing are smaller in this case.

[9] A diagram particularised to one parameter.

[10] These and the rest of the operations for the relationships defined in the model can be studied in more detail in [8].

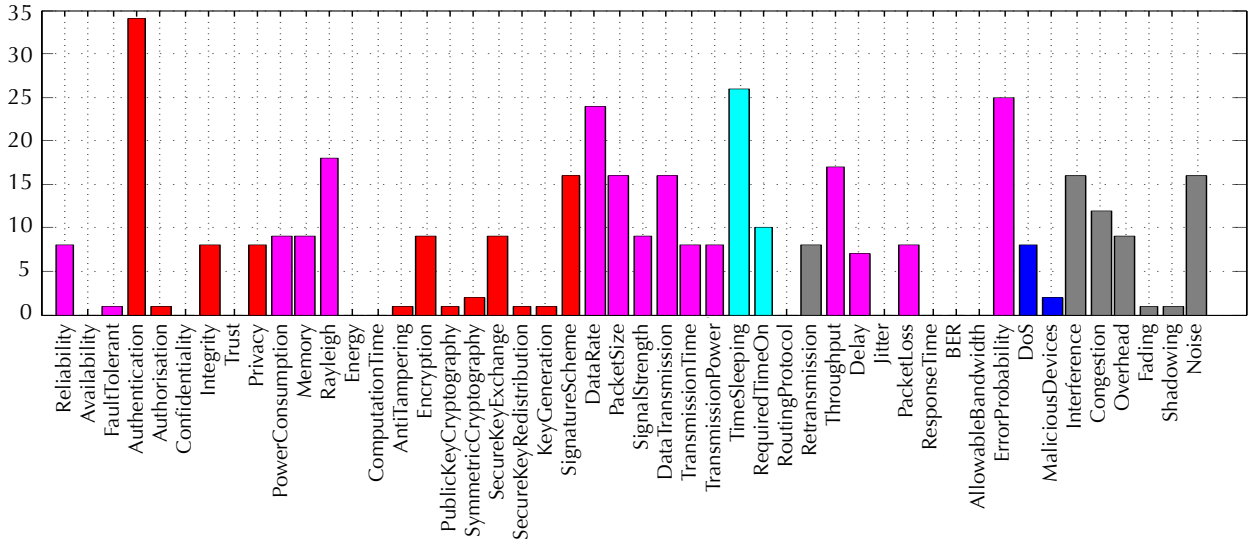


Figure 7: Dependence degree.

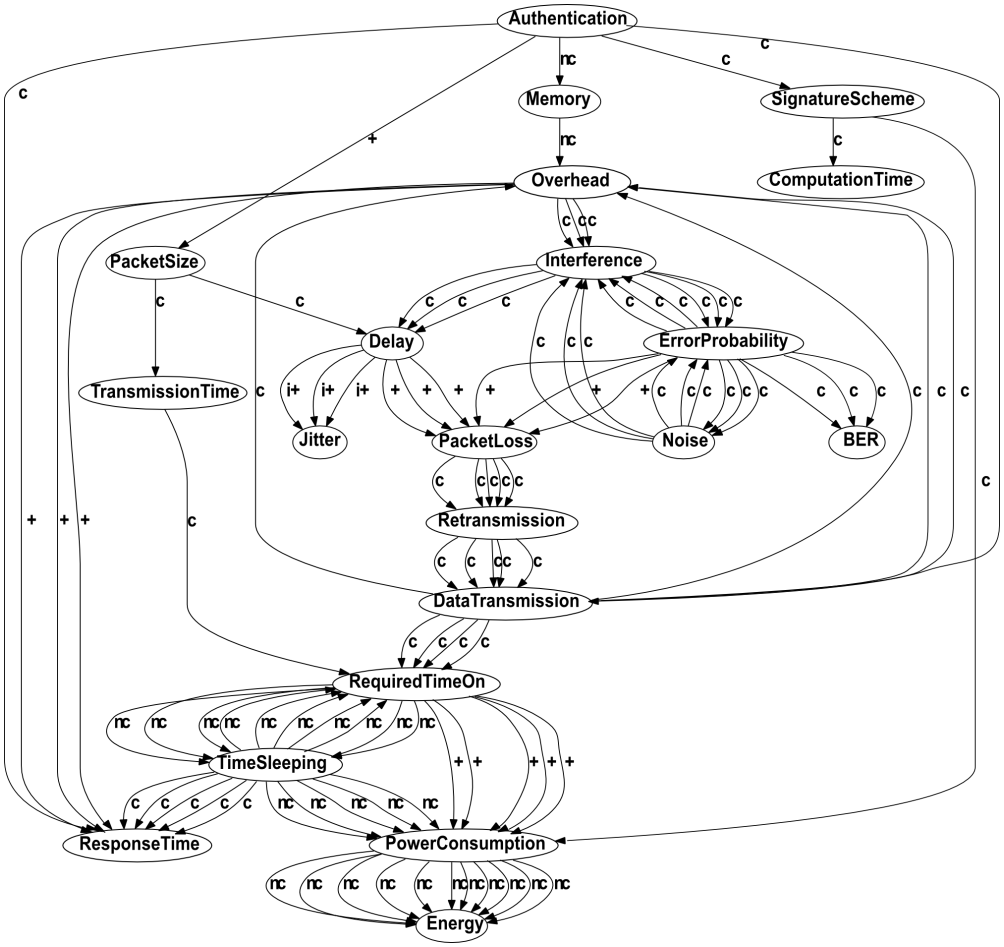


Figure 8: Increasing authentication.

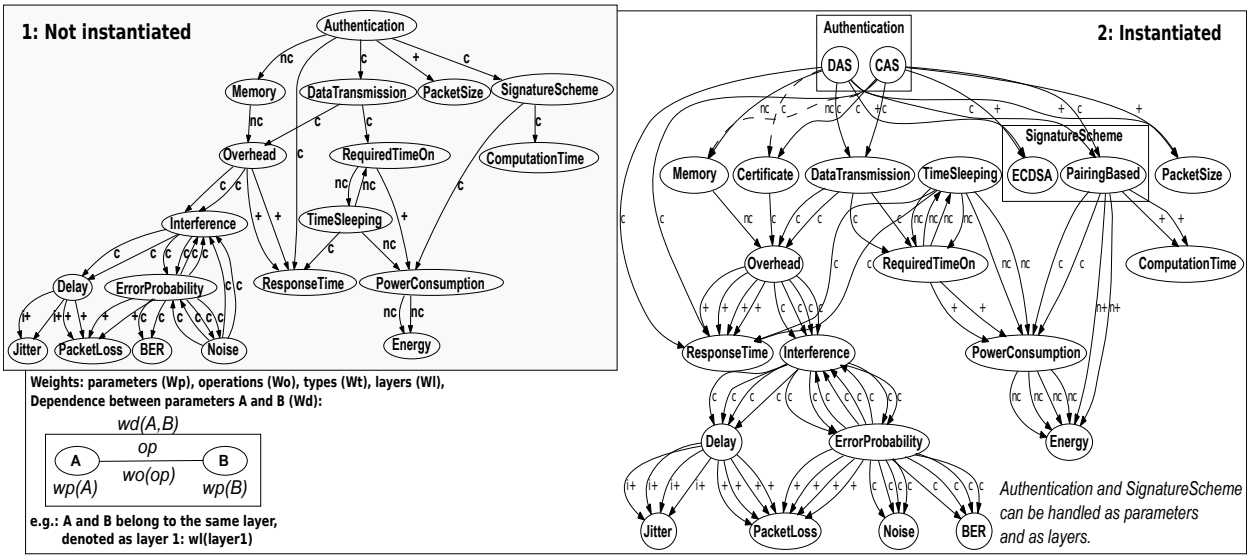


Figure 9: Decreasing Authentication: before & after the instantiation.

6.3. Anaysis of parameters, after the instantiation

Here we show the new tree for the parameters instantiated, that is, the parents. Using the PC defined in Section 5, and the given relationships, Figure 9 (case 2) shows the parametric tree for the parameter Authentication, once it has been instantiated. Moreover, as Authentication is related to SignatureScheme, then both instantiated parameters appear in Figure 9.

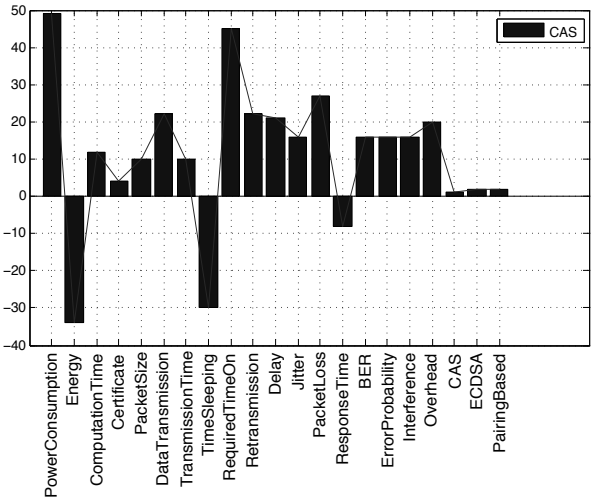
Note that, the relationships are duplicated because CAS and DAS are shown in the diagram, and both inherit the relationships from their parent, Authentication. Moreover, Authentication integrates the new behaviour defined by CAS, $CAS \rightarrow Certificate$. This new behaviour is integrated using weight 0, because according to rule AR3 any relationship for an instance must be supported by the instantiated parameter without affecting the predefined behaviour of the parent. Therefore, as the behaviour of Authentication is not modified by this new information, DAS inherits the new information but its behaviour is not affected by this new relationship because the weight is 0 (dashed line). This is due to the application of the rules in Table 1.

However, these similarities, do not affect the tradeoff analysis, which is performed based on the weights, that are not visible in the parametric tree diagram. Note that the weights defined in the PC for the relationships of CAS and DAS are different, so instead of inheriting the weights for the relationships of Authentication, both parameters take their own specific conditions and measurements. Therefore, the final impact of them on the final system is different, as Figure 10 shows. Indeed, the results after increasing the parameters CAS and DAS differ, particularly, in the set of parameters: PowerConsumption, PacketSize, DataTransmission, TransmissionTime, Delay, ResponseTime and Overhead. A common graph for both mechanisms is shown in Figure 11.

In this case, we consider it better to show the results for the increasing of both parameters, because it helps in the case that a system administrator wants to use SQT to assess the Security and QoS tradeoff, focusing on the two mechanisms used for implementing Authentication.

Remember that one of the objectives of DAS by avoiding the use of certificates is to reduce the overhead. However, in our scenario the value for Overhead is higher using DAS than using CAS. This is because we consider additional parameters in our analysis. Note that in our analysis, the parameter Memory affects the Overhead with weight 5. There are other parameters that also affect the Overhead, for example the PacsetSize. However, the final impact is higher using DAS than CAS. This can be different if the weight for the relationship $CAS \rightarrow Certificate$ is higher than the current value (2).

Moreover, note that the value of PowerConsumption in CAS is higher than in DAS, according to Figure 11. As can be noted in Figure 12, the



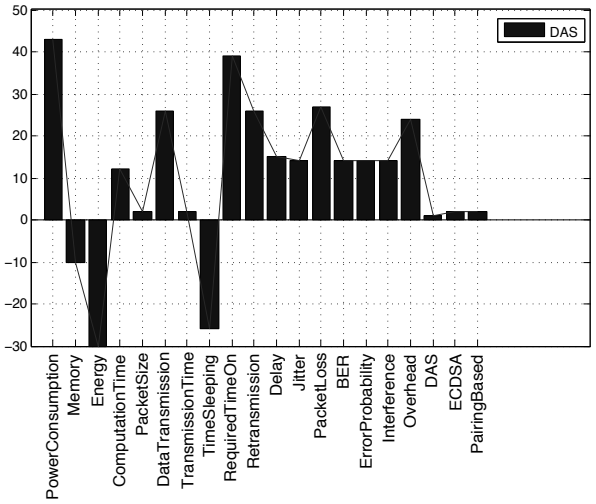


Figure 10: Impact of CAS and DAS on the Performance.

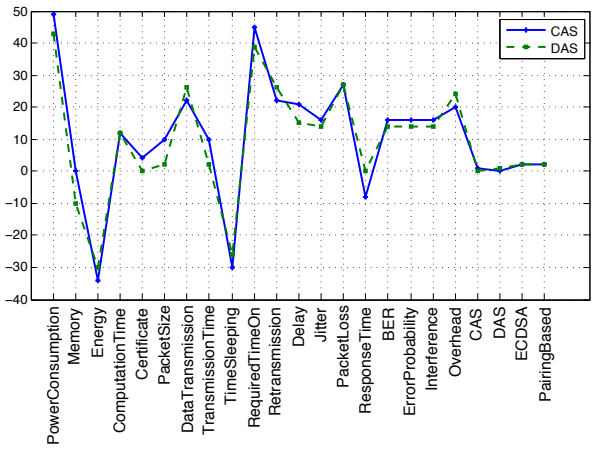


Figure 11: CAS versus DAS.

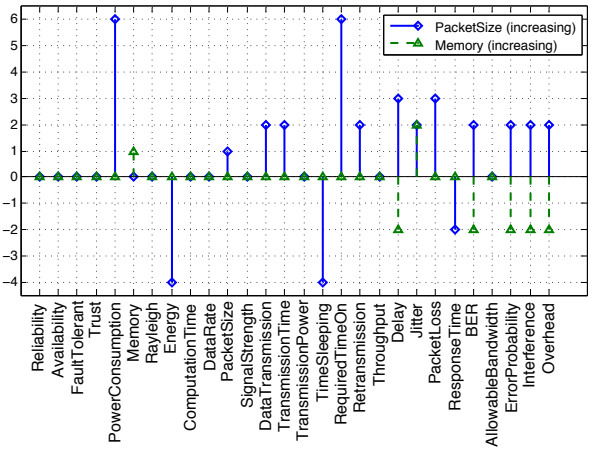


Figure 12: Packet size versus Memory.

increase in PacketSize affects PowerConsumption much more than the increase in Memory. Considering that the relationship $CAS \xrightarrow{c} PacketSize$ takes weight 5 (Table 5), the impact of this authentication mechanism on PowerConsumption is justified. Note that this conclusion is reached after considering the usual parameter set and the defined relationships.

Focusing on the main differences, in this use case, if we need to use CAS, PowerConsumption is a key value that has to be considered. Taking this into account, given the information in Table 5, which was set in the model, intuitivly CAS should be combined with ECDSA. However, it is possible to find new combinations in the intermediary parameters (in the parameter tree) in order to minimise the impact of CAS on PowerConsumption.

This is only an example, given a known use case, but the idea is that this type of formulation of a system based on parameters and the dependencies between them, enables the evaluation of different mechanisms under a common language. Therefore, if these results are combined, for example, with authorisation mechanisms, we can extract new combinations of mechanisms to improve the configuration of the final system.

Finally, note that, while the relationships in Table 5 were extracted from [18], because they provided a good example to prove our approach, in our analysis we show the effect that these mecha-

nisms can have on other parameters that have not been considered in Table 5. The idea behind SQT is precisely that we can combine different sources of information in order to evaluate the final set of data as a common behaviour, or context.

7. Conclusions and Future Work

Unified Communications (UC) will generate large amounts of data that can be processed to identify the dependencies between Security and QoS parameters and then, use these values to improve the final configuration of heterogeneous systems.

In this paper an extensive description of a Context-based Parametric Relationship Model (CPRM), to define Security and QoS parametric relationships in heterogeneous systems has been provided, and finally implemented. This has resulted in a tool for assessing the Security and QoS tradeoff (SQT) dynamically, based on the knowleadge collected from the environment. SQT provides an initial set of security and QoS parameters and their relationships, that can be enhanced by the user as needed. The usability of SQT has been tested, based on a use case, where the parameter Authentication was instantiated using two coexisting mechanisms for its implementation.

One improvement on the current solution could be the implementation of a recommendation system to help the user to analyse and reason about the final configuration decision. The most valuable improvement would be to increase the parameters and relationships to test our solution in complex scenarios. Furthermore, we think that an interesting point to solve in future work is how monitoring information could be automatically integrated from a productive system. To do so, the tools deployed have to be described using the syntax of CPRM, however, this automatic process could help avoid having to manually define the parameters, relationships and weights.

Acknowledgments

This work has been funded by Junta de Andalucia through the projects PISCIS (TIC-6334) and FISICCO (TIC-07223), and by the Spanish Ministry of Economy and Competitiveness through the project PERSIST (TIN2013-41739-R). The first author has been funded by the Spanish FPI Research Programme.

References

[1] J. F. Baxter Jr, Unified communication system, uS Patent 8,428,228 (Apr. 23 2013).

[2] A. Nieto, J. Lopez, Analysis and taxonomy of security/qos tradeoff solutions for the future internet, Security and Communication Networks 7 (12) (2014) 2778–2803.

[3] A. Aldini, M. Bernardo, A formal approach to the integrated analysis of security and qos, Reliability Engineering & System Safety 92 (11) (2007) 1503–1520.

[4] T. Taleb, Y. H. Aoul, A. Benslimane, Integrating security with qos in next generation networks, in: Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE, IEEE, 2010, pp. 1–5.

[5] A. Imran, A. Zoha, A. Abu-Dayya, Challenges in 5g: how to empower son with big data for enabling 5g, Network, IEEE 28 (6) (2014) 27–33.

[6] K. Park, Y. Kim, J. Chang, Semantic reasoning with contextual ontologies on sensor cloud environment, International Journal of Distributed Sensor Networks 2014.

[7] P. Sivakumar, K. Amirthavalli, M. Senthil, Power conservation and security enhancement in wireless sensor networks: A priority based approach, International Journal of Distributed Sensor Networks 2014.

[8] A. Nieto, J. Lopez, A context-based parametric relationship model (cprm) to measure the security and qos tradeoff in configurable environments, in: IEEE International Conference on Communications (ICC), IEEE, 2014, pp. 755–760.

[9] F. Martinelli, I. Matteucci, Partial model checking for the verification and synthesis of secure service compositions, in: Public Key Infrastructures, Services and Applications, Springer, 2014, pp. 1–11.

[10] F. Karatas, L. Fischer, D. Kesdogan, Service composition with consideration of interdependent security objectives, Science of Computer Programming 97 (2015) 183–201.

[11] H. Mostafa, N. Soule, N. Hoff, P. Pal, P. Hurley, Applying distributed optimization for qos-security tradeoff in a distributed information system, in: Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems, International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 1261–1262.

[12] S. Shin, P. A. Porras, V. Yegneswaran, M. W. Fong, G. Gu, M. Tyson, Fresco: Modular composable security services for software-defined networks., in: NDSS, 2013.

[13] M. Tasch, R. Khondoker, R. Marx, K. Bayarou, Security analysis of security applications for software defined networks, in: Proceedings of the AINTEC 2014 on Asian Internet Engineering Conference, ACM, 2014, p. 23.

[14] Y.-J. Chen, F.-Y. Lin, L.-C. Wang, Dynamic security traversal in openflow networks with qos guarantee, International Journal of Science and Engineering 4 (2014) 251–256.

[15] R. Di Pietro, S. Guarino, N. Verde, J. Domingo-Ferrer, Security in wireless ad-hoc networks–a survey, Computer Communications 51 (2014) 1–20.

[16] M. I. Khan, W. N. Gansterer, G. Haring, Static vs. mobile sink: The influence of basic parameters on energy efficiency in wireless sensor networks, Computer communications 36 (9) (2013) 965–978.

[17] M. Alia, M. Lacoste, R. He, F. Eliassen, Putting together qos and security in autonomic pervasive systems, in: Proceedings of the 6th ACM workshop on QoS and security for wireless and mobile networks, ACM, 2010, pp. 19–28.

[18] R. Yasmin, E. Ritter, G. Wang, An authentication framework for wireless sensor networks using identity-based signatures, in: Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on, IEEE, 2010, pp. 882–889.

[19] K. Ren, W. Lou, Y. Zhang, Multi-user broadcast authentication in wireless sensor networks, in: Sensor, Mesh and Ad Hoc Communications and Networks, 2007. SECON '07. 4th Annual IEEE Communications Society Conference on, 2007, pp. 223–232. doi:10.1109/SAHCN.2007.4292834.