Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 725279, 18 pages
http://dx.doi.org/10.1155/2014/725279

Research Article

Quality of Protection Evaluation of Security Mechanisms

Bogdan Ksiezopolski,[1,2] Tomasz Zurek,[1] and Michail Mokkas[2]

[1] *Institute of Computer Science, Maria Curie-Sklodowska University, Plac Marii Curie-Sklodowskiej 5, 20-031 Lublin, Poland*
[2] *Polish-Japanese Institute of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland*

Correspondence should be addressed to Bogdan Ksiezopolski; bogdan.ksiezopolski@acm.org

Received 10 March 2014; Revised 18 June 2014; Accepted 19 June 2014; Published 17 July 2014

Academic Editor: Fei Yu

Recent research indicates that during the design of teleinformatic system the tradeoff between the systems performance and the system protection should be made. The traditional approach assumes that the best way is to apply the strongest possible security measures. Unfortunately, the overestimation of security measures can lead to the unreasonable increase of system load. This is especially important in multimedia systems where the performance has critical character. In many cases determination of the required level of protection and adjustment of some security measures to these requirements increase system efficiency. Such an approach is achieved by means of the quality of protection models where the security measures are evaluated according to their influence on the system security. In the paper, we propose a model for QoP evaluation of security mechanisms. Owing to this model, one can quantify the influence of particular security mechanisms on ensuring security attributes. The methodology of our model preparation is described and based on it the case study analysis is presented. We support our method by the tool where the models can be defined and QoP evaluation can be performed. Finally, we have modelled TLS cryptographic protocol and presented the QoP security mechanisms evaluation for the selected versions of this protocol.

1. Introduction

Balancing security against performance in IT systems is one of the important issues to be solved. The traditional approach assumes that the best way is to apply the strongest possible security measures, which makes the system as secure as possible. Unfortunately, such reasoning can lead to the overestimation of security measures which causes an unreasonable increase in the system load [1, 2]. This problem is especially important in multimedia systems. The example from [2] may relate to the audio/video streaming as the real-time service. This teleconference can be protected by the VPN data transmission which will be accomplished by the TLS tunneling. After the service requirements analyses one can assign different versions of the protocol depending on the level of protection. The selection is presented in Table 1. In the first version one chooses the RC2-CBC algorithm and MD5 hash function. In the second version one selects the strongest symmetric algorithm (DES-CBC) and hash function with the longest digest (SHA1). In the third version one selects the symmetric algorithm with the key 3DES-CBC longer than the one selected in the second version.

In [2] authors checked how the security mechanism influences the efficiency of the peers during the video teleconference. The speed of transmitting data in the video conference which was secured by the VPN connection was checked. The results are presented in Table 2. The required quality of the video conference can be guaranteed only if we can transmit 240 KB/s and simultaneously receive the same amount of data. The transfer of the required level (480 KB/s) is guaranteed by the first version of the protocol (low). For the second version (medium) it is equal to the required bit rate. The third version of the protocol, which accomplished the VPN connection on the high level, cannot make the video conference of the required quality. The presented results show that overestimation of security mechanisms during data transmission leads to the decreasing efficiency of the devices from which the transmission is accomplished.

The system performance is also important in the systems with limited resources, such as wireless system networks

Table 1: Selection of the specific cryptographic algorithms.

Ciphers
Version 1—low
RC2-CBC + MD5
Version 2—medium
DES-CBC + SHA1
Version 3—high
3DES-CBC + SHA1

Table 2: The bit rate of VPN connections.

Version 1—low	
Bit rate	501 KB/s
Version 2—medium	
Bit rate	480 KB/s
Version 3—high	
Bit rate	453 KB/s

or mobile devices. Another example where such an analysis should be performed is cloud architecture. The latest research indicates that the three main barriers for using cloud computing are security, performance, and availability [3]. Unfortunately, when the strongest security mechanisms are used, system performance decreases and system availability is further influenced. The solution is using the QoP models. The latest results show [2, 4–6] that in many cases the better way is determination of the required level of protection and adjustment of some security measures to these requirements. Such an approach is achieved by means of the quality of protection models where the security measures are evaluated according to their influence on system security.

One of the most challenging issues in all QoP models is performing quality of protection evaluation of security mechanisms for the different versions of the cryptographic protocol (security policies). All of the approaches [4, 7–9] introduce different formulae which estimate the influence of security mechanisms for QoP, but they also have one significant limitation. These models can evaluate only these versions which were previously directly defined and described in an evaluation system. As directly defined scenarios we understand previously predefined models of configurations of security mechanisms which secure the IT processes. Unfortunately, defining all possible scenarios for all IT processes is very complex and in many cases is not feasible. The result of a nondefined scenario can be the situation in which security mechanisms of a specific IT process would not be evaluated; for example, adding a new security mechanism to an existing system may entail its lack of adjustment to any existing scenario and, consequently, the failure of its evaluation. The QoP evaluation of security mechanisms can be performed as part of risk analysis process or can be part of the decision support system, which in an adaptable way define appropriate configuration of security mechanisms. As a result of the lack of QoP evaluation of security mechanisms, the decision support system would not perform action adequate to the situation. This limitation is especially important in real-time systems [2, 10].

In the presented paper, we introduce a model of QoP evaluation of security mechanisms. Our main contribution is that the QoP evaluation of security mechanisms can be performed for not directly defined configurations of security mechanisms. The additional contribution of the paper is implementing the security mechanisms evaluation tool (SMETool) which supports the presented method. This tool can be used either by researchers or by security engineers. The SMETool can be downloaded from the web page of the Quality of Protection Modelling Language Project [11].

The paper is organized as follows. In the second section the related work about QoP models is presented. In the third section the formal model definition is presented. In the fourth section the methodology of QoP evaluation of security mechanisms is described. The fifth section deals with the case study of QoP evaluation of security mechanisms, where the TLS handshake protocol is presented. Finally, section six includes the conclusions.

2. Related Work

In the literature the security adaptable models are introduced as the quality of protection (QoP) models [4, 7–9, 12–17]. These models were created for different purposes and have different features and limitations. The related research in this area is presented below.

Lindskog attempt to extend the security layers in a few quality of service (QoS) architectures [14]. Unfortunately, the descriptions of the methods are limited to the confidentiality of data and based on different configurations of the cryptographic modules. Ong et al. in [15] present the QoP mechanisms, which define security levels depending on security parameters. These parameters are as follows: key length, block length, and contents of an encrypted block of data. Schneck and Schwan [16] propose an adaptable protocol concentrating on the authentication. By means of this protocol, one can change the version of the authentication protocol which finally changes the parameters of the asymmetric and symmetric ciphers. Sun and Kumar [17] create the QoP models based on the vulnerability analysis which is represented by the attack trees. The leaves of the trees are described by means of the special metrics of security. These metrics are used for describing individual characteristics of the attack. In [4] Ksiezopolski and Kotulski introduce mechanisms for adaptable security which can be used for all security services. In this model the quality of protection depends on the risk level of the analysed processes. Luo et al. [8] provide the quality of protection analysis for the IP multimedia systems (IMS). This approach presents the IMS performance evaluation using Queuing Networks and Stochastic Petri Nets. LeMay et al. [13] create the adversary-driven, state-based system security evaluation, the method which quantitatively evaluates the strength of systems security. In [9] Petriu et al. present the performance analysis of security aspects in UML models. This approach takes as an input of a UML model of the system designed by the UMLsec extension [18] of the UML modelling language. This UML model is annotated with the standard UML profile for schedulability, performance, and time and then analysed for performance. In [12] Ksiezopolski introduce the quality of protection modelling language (QoP-ML) which provides the modelling language for making abstraction of cryptographic protocols that put emphasis on the details concerning quality of protection. The intended use of QoP-ML is to represent the series of steps which are described as a cryptographic protocol. The QoP-ML introduced the multilevel [19, 20] protocol analysis that extends the possibility of describing the state of the cryptographic protocol. In [7] the authors present the impact of used security mechanisms in wireless local area networks for its quality of service. In this approach the QoP model is introduced which quantifies the benefits of security policies and demonstrates the relationship between QoS and QoP.

3. Model

Main aim of the model is to create a tool which helps to evaluate the quality of security protection of a given IT system. We are going to achieve it by evaluation of a set of security attributes of a given system, where as security attributes we understand various aspects of protection of a given system. At the beginning of the evaluation process we have a system which can be described by a set of facts. These facts represent all the elements of a given system. On the grounds of a set of facts we may, based on knowledge base, knowledge representation mechanism, and expert system-like forward chaining mechanism, infer a more general description of a given system and perform evaluation of the quality of protection of the analyzed system. The model is a semiformal tool which should allow to represent features of the analyzed system, as well as knowledge required to perform evaluation of the quality of protection of the system. The model also assumes the utilisation of the inference mechanisms which are a slightly modified version of expert systems like forward chaining mechanisms.

The goals of the presented model are as follows.

(1) The QoP evaluation of security mechanisms can be performed for not directly defined scenarios.

(2) Making quality of protection evaluation of all security mechanisms possible.

(3) Analysis refers to all security attributes.

(4) The model can be used for any QoP models.

3.1. Facts and Rules. We assume that the system behaviour modelled in one of the QoP models is represented by means of a set of propositions which we call facts:

$$F = \{f_1, f_2, f_3, \ldots, f_n\}, \tag{1}$$

where $f_1, \ldots, f_n$ are the facts describing a system.

By means of the facts one can define any security mechanisms. We assume a set of operators $\mathrm{OP} = \{\neg, \sim, \vee, \wedge, \Rightarrow, \rightarrow\}$, where

(i) $\neg$ is a classical (strong) negation;

(ii) $\sim$ is a negation as failure;

(iii) $\vee$ is a disjunction;

(iv) $\wedge$ is a conjunction;

(v) $\Rightarrow$ is a defeasible implication;

(vi) $\rightarrow$ is a strict implication.

Definition 1 (literals). Facts (negated in a strict way or nonnegated) are literals. The set of all literals is $L = \{l_1, l_2, \ldots, l_m\}$.

For example if $F = \{f_1, f_2\}$ is a set of facts then $L = \{f_1, f_2, \neg f_1, \neg f_2\}$ is a set of literals.

Definition 2 (case). A case is a model of an evaluated system represented by a set of literals $C = \{l_a, l_b, \ldots, l_s, l_z, \ldots\}$, which may be also expressed by a set of positive or negated facts: $C = \{f_a, f_b, \ldots, \neg f_s, \neg f_z, \ldots\}$.

Definition 3 (security attribute). Security attribute is an attribute which describes system behaviour in the case of information security requirements.

For example, one can enumerate the following security attributes [4, 21, 22]: integrity, confidentiality, authentication, availability, or anonymity. The security attributes (SA) set consists of an unlimited but finite number of security attributes. Each of them has its own evaluation value expressed by a positive integer number. Evaluation value represents the estimation of its security attribute. A security attribute may have a positive or negative character, in the sense that bigger value of security attribute evaluation may mean better (for positive) or worse (for negative) evaluation.

Definition 4 (rule). Rule is a formula in the following form:

$$\text{Conditions} \longrightarrow \text{Conclusion}, \tag{2}$$

where

(i) Conditions is a list of rule conditions.
A list of conditions is in the form wl_a *func* wl_b *func* $\cdots$ wl_d, where *func* is one of the operators from the set $= \{\vee, \wedge\}$, and $\{wl_a, wl_b, \ldots, wl_d\}$ are the facts (nonnegated or negated by negation as failure). Only one kind of operators can be used in one rule.

(ii) Conclusion is a rule conclusion in the form $\text{Conclusion} = (lx \wedge ly \wedge \cdots)$, where $(lx \wedge ly \wedge \cdots) \in L$.

Conditions may be negated by negation as failure and conclusions may be negated by classical negation. In the antecedent part of the rule, it is forbidden to use classical negation. In the consequent part of the rule, it is forbidden to use negation as failure. The set of rules is denoted as *RF*.

Rules allow us to represent relations between various facts, because in real life systems the existence of a chosen feature causes the existence (or not) of some other features. They also allow us to express which facts are exclusive in the sense that the existence of one of them causes the nonexistence of others. It is important because it helps to preserve consistency of the model.

3.2. Evaluation Rules. The assessment of the security attributes of a given computer system is based on facts and evaluation rules.

Definition 5 (evaluation rule). Evaluation rules are formulae in the following form:

$$\text{Conditions} \Longrightarrow \text{Inf}^{V}(\text{sa}), \tag{3}$$

where

(i) Conditions is a list of rule conditions in the form wl_a *func* wl_b *func* $\cdots$ wl_d, where *func* are the operators from the set $= \{\vee, \wedge\}$ and $\{wl_a, wl_b, \ldots, wl_d\}$ are facts (nonnegated or negated by negation as failure).

(ii) Inf is a function changing value of evaluation of security attribute sa by adding value V (security influence) to the security attribute evaluation, when V is an integer number which represents the evaluation of a given security attribute.

Security attribute evaluation cannot be lower than 0. Special function $\text{Inf}^0(\text{sa})$ means that the security attribute sa evaluation is reduced to 0. This function will reduce sa to 0 regardless of sa, the current value. When this operator is used, it means that this sa is not guaranteed. The V value of sa will be equal to 0 disregarding other evaluation rules which could increase V value. This mechanism will be described more precisely later.

It is important to notice that in fact evaluation rules are not rules in a traditional sense of this term. They are conditionals in which the satisfaction of their conditions causes change of value of evaluation attribute.

We denote the set of evaluation rules as a ER. The example of the evaluation rules set ER is

$$\begin{gathered} f_1 \wedge f_2 \Longrightarrow \text{Inf}^{10}(\text{Confidentiality}), \\ f_3 \wedge \sim f_4 \Longrightarrow \text{Inf}^{0}(\text{Integrity}). \end{gathered} \tag{4}$$

The fulfillment of the evaluation rule conditions causes an appropriate change of the security attribute evaluation value. For example, when we have the above rules and we have the case $P = \{f_1, f_2, f_3\}$, we may conclude that the value of evaluation of security attribute Confidentiality should increase by 10 points and the value of evaluation of security attribute Integrity should decrease to 0 points.

The evaluation rule uses defeasible implication because such a rule may be defeated by another one.

Definition 6 (strict satisfaction of rule conditions). Rule conditions are satisfied in a strict way if positive (nonnegated) conditions are true and negative conditions (facts negated with negation as failure) are false or it is impossible to conclude that they are true.

The example of the rule is

$$f_1 \wedge f_2 \wedge \sim f_3 \longrightarrow f_4. \tag{5}$$

From the above rule we may conclude that if f_1 and f_2 are true and f_3 is false (it is not declared, it cannot be concluded from other rules, and it is declared that $\neg f_3$ or there is a rule with the conclusion $\neg f_3$ and its conditions are satisfied), then f_4 is true.

3.2.1. Orders between Facts. It is easy to notice that the evaluation of complex systems requires building of a large set of rules, which should allow for evaluation of any real life system. It is also possible that such a set of rules may be, due to many reasons, incomplete in the sense that there may be facts which are not used in any rule and evaluation rule.

Such a situation may lead the evaluation process to misleading consequences which come from the lack of evaluation of potentially important facts. We can also notice that such new facts unpredicted in the rule base may be in a way connected to the other ones which are already regulated. They may, for example, represent better satisfaction of a condition of a chosen rule. On the other hand it is sometimes much easier to declare that, for example, fact f_1 means more than f_2 and may satisfy the condition of a chosen rule in a better way.

Based on the above, we assume the possibility of the declaration of orders between facts. The partial order $f_1 > f_2$ denotes that f_1 means more than f_2 and if there is a rule in which f_2 is one of the conditions and we know that f_1 is satisfied, then we may conclude that f_2 should also be satisfied (even if it is not literally true). Such an order represents better satisfaction of the rule condition. For example, if f_2 means cipher with the default key length, f_1 may denote cipher with the longer key length.

It is worth mentioning that these relations may not be the same for every security attribute. For example, a longer key length is better in the matter of confidentiality, but it is worse in the matter of efficiency. According to the above, we have to add to the order additional information about security attribute in which this reasoning concerns.

We also assume that the relation of order between facts is transitive:

$$\forall_{(X,Y,Z)} ((X > Y) \wedge (Y > Z) \longrightarrow (X > Z)). \tag{6}$$

We introduce the structure OF: $OF = \langle F, >_{\text{SA}} \rangle$, where $>_{\text{SA}}$ is a relation of strict partial order which represents preferences between various facts from the set F in the context of security attribute SA ($f_1 >_{\text{sa}} f_2$ denotes that f_1 means more than f_2 in the context of security attribute sa).

Definition 7 (unstrict satisfaction of rule conditions). Condition f_x of a given rule is satisfied in an unstrict way when f_x is not true, but

(i) there is a fact f_y;

(ii) it is not known that $\neg f_x$ ($\sim \neg f_x$);

(iii) we know that $f_y >_{\mathrm{SA}} f_x$;

(iv) reasoning concerns evaluation of security attribute SA.

Such kind of satisfaction of the condition of the rule we call unstrict satisfaction of rule conditions. The root of unstrict satisfaction of the rule lies in the so called a'fortiori reasoning (reasoning from more to less), which is commonly used in legal domain. The example of formalisation of such a way of reasoning is presented in [23].

Looking more generally at the above, we may notice that falsehood of the condition is not sufficient to assume that it is not satisfied. In other words, we have to distinguish truthfulness of the condition from its satisfaction. In the model we assume that false condition may be, in the above mentioned cases, treated as satisfied one. We also assume that it is not possible to treat true condition as unsatisfied.

Another important thing which is connected to the above defined unstrict satisfaction of the rule conditions is a necessity of preservation of the consistency of the model of the system (as consistency we understand here the exclusion of the possibility of existing complementary facts, for example, $f_1, \neg f_1$). The second clause of the above definition ($\sim \neg f_x$) controlling condition may be satisfied in the unstrict way only if it is not known that it is false and it is impossible to derive that it is false. This clause determines that unstrict satisfaction may be defeated by strict declaration of another fact or by another rule.

Definition 8 (satisfaction of the conditions of the rule). If there is a given case C described by a set of literals $C = \{l_x, l_y, \ldots, l_z\}$ which satisfies in a strict or unstrict way conditions (Conditions) of a rule $rf \in RF$, then we denote it as $C\bullet$ Conditions.

3.3. Inference Rule.

The above system has one important feature: there is a distinction between truthfulness of the condition and its satisfaction. In order to create an inference mechanism for our model, we have to modify the classic *Modus Ponens* rule.

Definition 9 (inference rule). As an inference rule we understand the following rule:

$$\frac{(\text{Conditions} \longrightarrow \text{Conclusions}) \wedge f \bullet \text{Conditions}}{\text{Conclusions}}, \tag{7}$$

where → is a strict implication and $f\bullet$ Conditions mean that fact f satisfies (in a strict or unstrict way) conditions of a given rule.

Strict or unstrict satisfaction of the rule antecedents allows us to treat rule conclusion as true and, consequently, also satisfied.

3.4. Inference Mechanism.

On the basis of the above defined inference rule, we have to define our inference mechanism.

Definition 10 (fact based inference mechanism). As a fact based inference mechanism we understand forward chaining mechanism using the inference rule defined earlier. As C' we denote a set of conclusions whose inference mechanism concludes from a case C, a set of rules RF and a set of orders OF. We may also denote it as $C \vdash C'$. The union of sets $C \cup C'$ we call a complete description of the case and denote as P.

3.5. Security Attributes.

As described above, the security attributes' set SA consists of an unlimited but finite number of security attributes. Each of them has its own evaluation value expressed by a positive integer number.

Definition 11 (set of security attributes pairs.). S is a set of pairs $O = \langle \mathrm{sa}, o \rangle$, where $\mathrm{sa} \in \mathrm{SA}$ is a security attribute and o is its evaluation value.

For example, we have three security attributes with their evaluation values:

$$\mathrm{SA} = \text{confidentiality, integrity, authorisation;}$$

$$S = \{(\text{confidentiality}, 10), (\text{integrity}, 20), (\text{author.}, 30)\}. \tag{8}$$

It is important to notice that security attributes may have positive or negative character, in the sense that a higher value of security attribute evaluation may mean better (for positive) or worse (for negative) evaluation.

3.6. Conflicts between Rules.

Some specific conditions may cause conflicts between evaluation rules.

Definition 12 (conflicting rules). There is a conflict between two or more evaluation rules if these rules cannot be executed together.

Such conflicts may appear when there are two rules whose antecedents are satisfied, who are in a way connected and the execution of both of them may cause improper influence on the security attribute evaluation.

The problem of conflicting and subsuming rules is the main reason for the utilisation of defeasible implication. In this work as defeasibility of the evaluation rules we understand the possibility of exclusion from the evaluation process of a chosen rule by another rule. If antecedents of two conflicting rules are satisfied, only one of them may be executed (but such a rule may be also defeated by another one).

To represent priorities between the evaluation rules we assume partial order between rules from a set ER. Such an order allows us to express that if $r_1 > r_2$ and $r_1, r_2 \in ER$, then rules r_1 and r_2 are in conflict and when the conditions of both of these rules are satisfied rule r_1 should defeat rule r_2.

3.6.1. Reasoning about Orders between Conflicting Rules.

The main problem of the above mentioned issue of conflicting rules lies in the mechanism of recognition of conflicting rules. Generally, such recognition should be based on commonsense reasons, but there is one phenomenon, which allows us to recognize conflict and to find order between conflicting rules. This situation concerns a case where there are two rules, one of which has a more general set of conditions than the other one. In other words, every case satisfying condition of the rule r_2 also satisfies conditions of the rule r_1. Such subsumption of the rules allows us to conclude that the rule r_2 is a specific case of the rule r_1, and in the case of satisfaction of the antecedents of both rules, the rule r_2 should defeat the rule r_1. More generally we may say that two rules are in conflict when the list of conditions of one of them subsumes that of the second one and the conclusions of both concern the modification of the same security attribute evaluation value.

Definition 13 (subsuming rules). When we have the following two rules:

$$\begin{aligned} r_x &: \text{Condition}_x \Longrightarrow \text{Inf}^{V1}(\text{sa}), \\ r_y &: \text{Condition}_y \Longrightarrow \text{Inf}^{V2}(\text{sa}), \end{aligned} \tag{9}$$

where Condition_x and Condition_y are the lists of antecedents of these rules, both rules influence the same security attribute evaluation value sa (but they may have a different level of influence), and if for any case P represented by a set of literals:

$$\forall_{P\in L}\left(\left(P \bullet \text{Condition}_x\right) \longrightarrow \left(P \bullet \text{Condition}_y\right)\right), \tag{10}$$

then we recognise the rules r_x and r_y as subsuming and conflicting ones and in the view of a more restrictive character of the rule r_x we may conclude that the rule r_x has priority over the rule r_y, which we denote: $r_x > r_y$ and while conditions of both rules are satisfied, the rule r_x should defeat the rule r_y.

Rules with the function $\text{Inf}^0(\text{sa})$ on the consequent part of the rule have the highest possible priority and they are in conflict with all the rules concerning the security attribute sa. Everytime when they satisfy conditions, they exclude all evaluation rules concerning the security attribute sa from reasoning.

Another aspect which is important and requires explanation is connected with the reason why a more specific rule defeats a more general one. Such a mechanism comes from the theory of law and is called *lex specialis derogat legi generali*. It is one of the tools which allow to find a solution in conflicting legal rules, saying that a specific act (provision) derogates from (prevails over) the general regulation. In modelling legal rules there are many problems connected with the difficulties with recognition which of the rules are more or less specific [24]. In the case of our model it is much simpler because the hierarchy of generality of antecedents of the rules is easy to establish based on a finite number of possible facts and their explicitness.

3.7. Evaluation Rules System

Definition 14 (evaluation rules system). Evaluation rules system RO is a structure described by a set of evaluation rules ER and relation OR= $\langle \text{ER}, > \rangle$.

The relation OR represents partial order between rules from a set ER. This order maps preferences between conflicting rules. These preferences can come from strict declaration or from a previously defined mechanism of finding and resolving a problem of subsuming rules. If there is a relation of partial order between two rules, we treat them as conflicting ones. If these rules are not comparable, we treat them as conflict free. We also assume that the relation of order between rules is transitive:

$$\forall_{r_x,r_y,r_z}\left(\left(r_x > r_y\right) \land \left(r_y > r_z\right) \longrightarrow \left(r_x > r_z\right)\right). \tag{11}$$

3.8. QoP Evaluation Process of Security Mechanisms. The process of QoP evaluation of security mechanisms is expressed by means of evaluation of security attributes. The values of the SA depend on the used security mechanisms which are represented in the model by facts *F*.

The evaluation process of security mechanisms may be described in terms of a sequence of steps as presented by Algorithm 1. The parameters and variables used in this algorithm are presented in Table 3.

3.9. Background of the Model. Looking more generally at our model, one can notice that it contains some elements taken from the formal models of legal reasoning. Legal reasoning has a very specific character, as it requires mechanisms of dealing with incomplete knowledge, mechanisms of resolving conflicts between legal rules and arguments, various ways of interpretation of rules, and so forth. Computer systems which aims to support human reasoning very often, have to face similar problems, which is the main reason for the utilisation of the *a'fortiori* rule or the *lex specialis*... rule in our system. Similar models of legal reasoning have also been applied in other computer science utilisations; for example, in [25] the authors are forced to implement one of the methods of resolving the conflicts between rules in multiagent systems.

Our model is based on proposition logic but there are some additions which allow for a better representation of specific features of the analysed problem. First of all, we introduce a distinction between two kinds of negation: *classical negation* (strong) and *negation as failure*. As negation as failure we understand the negation used in the conditional part of the rule. Such a negated condition is fulfilled if it is impossible to satisfy such a condition (it is false, it is not declared, or it is impossible to derive that it is satisfied). These two kinds of negation are used in a few logical systems, for example, in the Prakken and Sartor logic [26] or the Kowalski and Toni logic [27]. In our model the way of the utilisation of negation as failure is similar to the one presented by Prakken and Sartor in [26], but we do not allow to use construction like $\sim \neg P$, because in our model it is forbidden to use negation as failure in the consequence part of the rule and it is also

```
SET $C$
for $i = 1$ to $n$ do
    $o_i \leftarrow 0$
    SET $OF[i]$
    SET $C'[i] = \text{RES}(C, OF[i], R)$
    $P[i] = C[i] \cup C'[i]$
    SET $\text{ER}[i]$

    if $\text{ER}[i] = \emptyset$ then
        $o_i \leftarrow 0$
        CONTINUE
        for $k = 1$ to $\text{NER}[i]$ do
            for $m = 1$ to $\text{NER}[i]$ do
                if $(\text{er}[k][i], \text{er}[m][i] \wedge \text{er}[k][i] > \text{er}[m][i])$ then
                    EXCLUDE $\text{er}[m][i]$ from $\text{ER}[i]$
                end if
            end for
        end for
    end if

    for $l = 1$ from $\text{NER}[i]$ do
        if exists $\text{er}[l][i]$ in $\text{ER}[i]$ such that conclusion is $\text{Inf}^0(i)$ then
            $o_i \leftarrow 0$
            CONTINUE
        else
            READ $V[l][i]$
            $o_i = o_i + V[l][i]$
        end if
    end for
end for
```

Algorithm 1: Algorithm of security attributes evaluation.

Table 3: The parameters and variables for the security attributes evaluation algorithm.

SET	Make a choice indication
EXCLUDE	Excluding from the ER indication
READ	Reading indication
CONTINUE	Processing statement will be skipped
$\text{RES}(C, OF[i], R)$	The reasoning function based on a set of facts C and order of facts $OF[i]$ for the security attribute i and rules R (inference mechanisms)
$OF[i]$	Orders between facts referred to a security attribute i
C	A case expressed by a set of facts
C'	A set of facts obtained from the inference mechanism
$P[i]$	A full description of a case for a security attribute i
R	A set of rules
k, m, l	Indicates the current evaluation rule
o_i	The evaluation of ith security attribute
$\text{ER}[i]$	A set of evaluation rules with satisfied conditions for the security attribute i
$\text{NER}[i]$	The number of rules with satisfied conditions for the security attribute i
$\text{er}[x][i]$	The evaluation rule x for the security attribute i
i	The index of the current security attribute
n	The quantity of security attributes
$V[l][i]$	The value of the security influence of the security mechanisms represented by the evaluation rule l for the security attribute i

forbidden to use classical negation in the antecedent part of the rule.

Another important point in our model, the conception of orders between facts, is based on a simplified version of the a'fortiori reasoning (reasoning from more to less: if norm N1 obliging to do more is binding, then norm N2 obliging to do less is binding more). In our work, this way of reasoning has been slightly modified: when condition X of a rule r_1 is not satisfied literally, but there is a fact Y which satisfies this condition in a better way we may treat such a condition as satisfied. A more profound analysis and model of the a'fortiori reasoning can be found in [23].

The utilisation of defeasible implication in evaluation rules is another important feature of our model. The idea of distinction of two kinds of implication comes from the formal models of legal argumentation in which the problem of defeasibility of the rules is broadly discussed. In the Prakken and Sartor logic [26], all arguments are defeasible. Vreeswijk in his abstract argumentation system [28] uses two kinds of implication (material $\supset$ and defeasible $>$). He assumes that the utilisation of defeasible implication requires the definition of a separate defeasible inference rule. Hage in his reason based logic [29] looks at the problem of defeasibility of the rules from another point of view, stating that this is a problem of the applicability of the rule. The rule could have satisfied conditions, but cannot be not applicable due to, for example, a conflict with another rule. Another interesting model of argumentation which uses defeasible and strict implication is introduced by Kowalski and Toni in [27]. In their system, each defeasible rule r has a condition $\sim$ defeated(r) (where $\sim$ is negation as failure and defeated(r) is a predicate which denotes that this rule is defeated by another one) which allows defeating it if it remains in conflict with another rule r' and has lower priority than r'.

Defeasible rules in our model are slightly different from those in the above mentioned models. The most important point in our model lies in the fact that only evaluation rules (which are not rules in a strict meaning of this word) are defeasible. Such a rule can be defeated only if its conditions are satisfied; it is in conflict with another rule with satisfied conditions and has a lower priority over the other one. Such a defeated rule is excluded from the reasoning process.

Torre and Tan in [30] define a few types of defeasibility and the one used in our model is closest to the *overridden defeasibility* which formalizes the cancellation of a rule by another one.

The ways of dealing with conflicts between rules and orders between these rules are discussed in the aforementioned works by Prakken and Sartor (i.e., [26, 31]) where the authors introduce their formal model of legal argumentation. The notion of conflict between rules in our approach is different from the one presented in their works, but the way of resolving it by the declaration of orders between rules and the assumption of defeasibility of rules is similar to the Prakken and Sartor model. In the above mentioned Kowalski and Toni logic, two rules are in conflict when they have complementary conclusions and the conditions of both rules are satisfied. Their model also uses two kinds of implication as well as two kinds of negation, but the way of dealing with conflict is slightly different from the one in our system. They define a few new predicates which are used to denote that the condition of the rule r is satisfied (holds(r)), rule r is defeated (defeated(r)) or two rules are in conflict (conflict(r, r')).

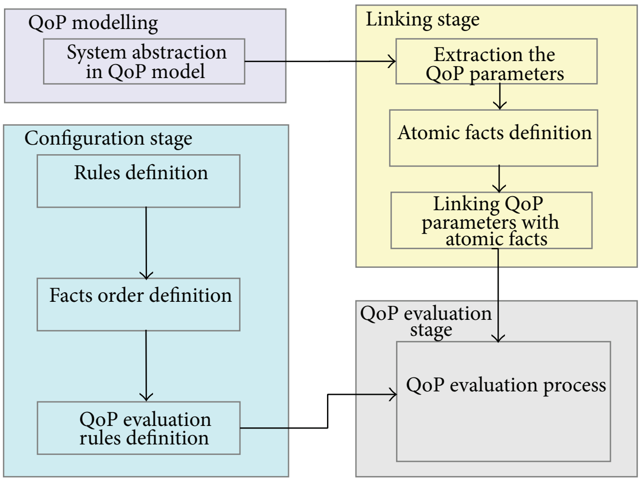

Figure 1: The methodology of QoP evaluation of security mechanisms.

4. Methodology of QoP Evaluation of Security Mechanisms

The QoP evaluation of security mechanisms is a process which can be divided into four stages: QoP modelling, linking, configuration, and QoP evaluation. In Figure 1 the methodology of the process is presented.

QoP Modelling Stage. The first stage refers to modelling the system in one of the QoP models [4, 7–9, 12–17]. This phase depends on the chosen QoP approach, but the result should be the same, and the system must be modelled with QoP meaning (*system abstraction in the QoP model*). The proposed model can be used for different QoP modelling approaches.

Linking Stage. The next stage is responsible for the linking system modelled in the QoP model with the structure defined in the proposed model. Firstly, one has to extract all parameters used in the QoP model which refer to the factors which influence QoP parameters (*extraction of the QoP parameters*). After this, one creates atomic facts in the proposed model which are explicit to these extracted from the model (*atomic facts definition*). The parameters' names used in the QoP model can be different from those defined in the model, so one has to link them together (*linking QoP parameters with atomic facts*).

Configuration Stage. The next stage is the main phase where all structures proposed in the model for QoP evaluation of security mechanisms are defined. One can enumerate: *rules definition*, *facts order definition,* and *QoP evaluation rules* *definition*. All of these structures are described in detail in Section 2.

QoP Evaluation Stage. The last stage is responsible for QoP evaluation of security mechanisms. The analysed specification can be defined directly in the QoP model and thanks to the previously defined links between the QoP parameter and facts, a model of a case is generated. Finally, after the system version indication, the QoP evaluation of security mechanisms is performed (*QoP evaluation process*).

5. Case Study: TLS Handshake Protocol

In this section we are going to present a case study of the QoP evaluation of security mechanisms for the TLS Handshake protocol. The TLS protocol is used each day in real business situations in the actual enterprise environment. Given the enterprise network infrastructure in Figure 2, one should analyse different roles which refer to different levels of the quality of protection of used security mechanisms. The users are allowed to access e-mail, FTP, web, and application servers with the communication channel protected by means of the TLS protocol at a different QoP level. The utilized versions of the TLS protocol together with equivalent cryptographic algorithms are summarized in Table 7.

The model for the TLS Handshake protocol can be found in the models library in the SMETool, which can be downloaded from the web page of the Quality of Protection Modelling Language Project [11]. In this case study, the client wants to verify the server and, next, to connect with the server by a secure connection. All of these security requirements can be realized by means of different security measures. We are analysing six versions of the protocol. The flow of the TLS Handshake protocol, which will be analysed further, is realized in five steps and the scheme is presented as shown in the following steps:

(1) $C \rightarrow S$: *ClientHello*

(2) $S \rightarrow C$: *ServerHello*, *Certificate*, *ServerKeyExchange*, *ServerHelloDone*

(3) $C \rightarrow S$: *ClientKeyExchange*, *ChangeCipherSpec*, *Finished*

(4) $S \rightarrow C$: *ChangeCipherSpec*, *Finished*

(5) $C \rightarrow S$: *Encrypted data.*

Below we present a description of these steps.

(1) A Client sends the *ClientHello* message. This message contains the following attributes: the TLS Protocol version, session id, a list of available cipher suite, compression method, and random values.

(2) The server responds with a *ServerHello*, which establishes the version of the TLS Protocol, session id (when session is resumed), cipher suite, and compression method. It also sends random values within *ServerHello*. Next the server sends the *Certificate* message which includes its certificate. Sending this message is not necessary; it depends on the selected cipher suite. The next message which may be sent is *ServerKeyExchange*. The server sends it when the server certificate is only for signing, or the server has no certificate. After this, the server sends *ServerHelloDone*, signalling that this phase is completed.

(3) Next, the client sends the *ClientKeyExchange* message. Depending on the selected cipher, this message may have different contents. After this, the *ChangeCipherSpec* message is sent. The client sends it to signal the server that it has started to use the encryption. Finally, the encrypted *Finished* message is sent by the client.

(4) Similarly, in response, the server sends *ChangeCipherSpec* and the encrypted *Finished* message.

(5) The handshake is complete. Now, the server and client can exchange *Encrypted data*.

The methodology of the QoP evaluation of security mechanisms based on our model is presented in Section 3. In this section we have used the proposed methodology for analysing the TLS Handshake cryptographic protocol.

5.1. QoP Modelling.

In the first step one has to model the system in one of the QoP models. In the presented case study, the TLS protocol is analysed as a full system. The QoP modelling process is a complex task so we are not going to present it in this paper. The example of the QoP model of TLS cryptographic protocol is presented in [6], where the protocol is modelled in the QoP-ML modelling language [12].

5.2. Linking Stage

5.2.1. Extraction of the QoP Parameters.

The first step in the linking stage refers to the extraction of all parameters used in the QoP model which refer to the factors which influence QoP. The form of the QoP parameters depends on the chosen QoP model. For example, in the QoP-ML modelling language, the system behaviour is changed by the functions which modify the states of the variables and pass the objects by communication channels. This structure is responsible for indicating the parameters which influence the system QoP.

Let us assume that qp is the QoP parameter in the QoP model which influences system security. The QoP parameter is atomic, which means that it cannot be split into other atomic QoP parameters. The QP is a set of the QoP parameters:

$$QP = \{qp_1, qp_2, qp_3, \ldots, qp_n\}, \qquad (12)$$

where

$qp_1, qp_2, qp_3, \ldots, qp_n$are the QoP parameters which influence system security;

QP is a set of the QoP parameters.

5.2.2. Atomic Facts Definition.

In the next step, we declare a set of all possible facts F which will represent the features of the analysed systems.

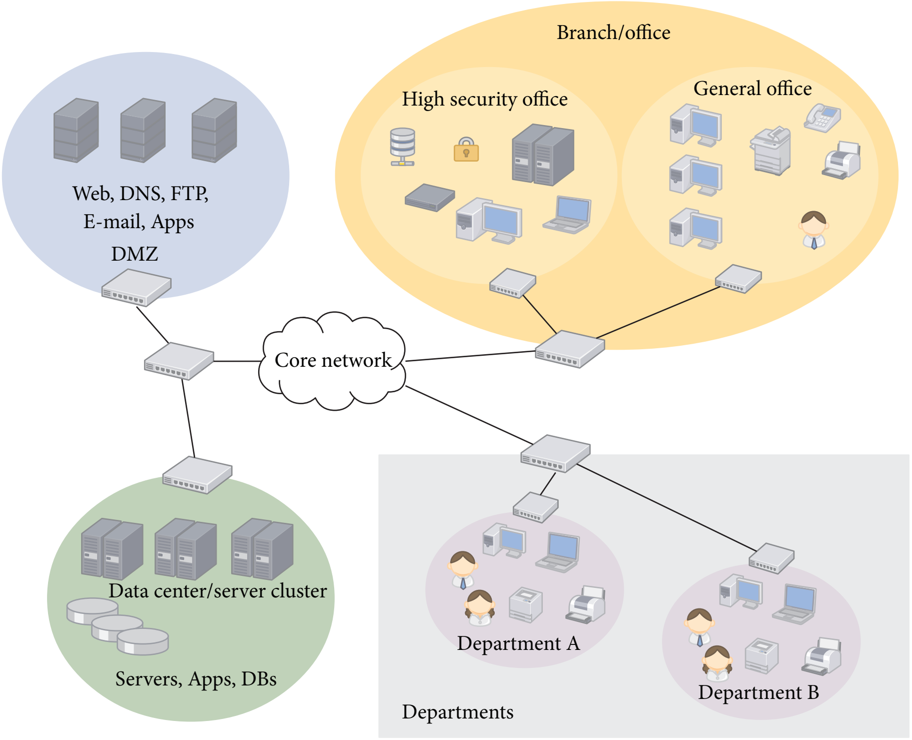

Figure 2: Example enterprise network architecture.

Table 4: The group of facts refers to symmetric encryption.

Facts
Group name: cipher (symmetric encryption algorithm)
f_1(cipher) = RC4
f_2(cipher) = 3DES
f_3(cipher) = AES
Group name: bs (block size in bytes)
f_1(bs) = 8
f_1(bs) = 16
Group name: IV (initiate vector in bytes)
f_1(IV) = 8
f_1(IV) = 16
Group name: key (key length in bytes)
f_1(key) = 16
f_2(key) = 24
f_3(key) = 32

In the presented case study we include the TLS cryptographic protocol. In the following section we define the atomic facts in the proposed model for the TLS protocol. These facts are taken from the official specification of the TLS protocol [32] where all possible versions of the protocol are defined. The facts are divided into three groups which refer to different factors: symmetric encryption (Table 4), message digest (Table 5), asymmetric encryption, and common facts (Table 6).

Table 5: The group of facts refers to message digest.

Facts
Group name: mac (message authentication code algorithm)
f_1(mac) = HMAC-MD5
f_2(mac) = HMAC-SHA1
f_3(mac) = HMAC-SHA256
Group name: mac-len (message digest length in bytes)
f_1(mac-len) = 16
f_2(mac-len) = 20
f_3(mac-len) = 32
Group name: k-len (mac key length in bytes)
f_1(k-len) = 16
f_2(k-len) = 20
f_3(k-len) = 32

5.2.3. Linking QoP Parameters with Atomic Facts. The names of the QoP parameters used in the QoP model can be different from atomic facts defined in the proposed model. In this step, the QoP parameters are linked with the facts in the proposed model in an explicit way.

Definition 15 (linking operator). The linking operator $\mapsto$ denotes that one set of objects is mapped to another set of objects in an explicit way.

As we assume that the *QP* is the set of the QoP parameters used in the QoP model for the TLS protocol abstraction

Table 6: The group of facts refers to asymmetric cryptography and common facts.

Facts
Group name: PK (key exchange algorithm scheme)
f_1(PK) = RSA
f_2(PK) =DH-DSS
f_3(PK) = DH-RSA
f_4(PK) = DHE-DSS
f_5(PK) = DHE-RSA
f_6(PK) = DH-anon
Group name: mode (mode of operation)
f_1(mode) = CBC
Group name: com (bit compression before encryption)
f_1(com) = Compression

Table 7: The analysed versions of TLS protocol.

Version	The cipher suite
1	TLS_RSA_WITH_RC4_128_MD5
2	TLS_RSA_WITH_AES_128_CBC_SHA
3	TLS_DH_anon_WITH_RC4_128_MD5
4	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
5	TLS_RSA_WITH_RC4_128_newSHA512
6	TLS_RSA_WITH_RC4_128_MD5 + COM

and F is the set of all atomic facts defined in the proposed model. Usually, the set of the QoP parameters QP is a subset of the facts in the model F, but in a special case this set can be equal to

$$QP = \{qp_1, qp_2, qp_3, \dots, qp_n\};$$
$$F = \{f_1, f_2, f_3, \dots, f_n\}; \quad (13)$$
$$Z \in F,$$

where

$qp_1, qp_2, qp_3, \dots, qp_n$ are QoP parameters which influence the system security;

QP is the set of QoP parameters;

$f_1, f_2, f_3, \dots, f_n$ are facts in the formal model;

F is the set of all facts in the model;

Z is the subset of all facts in the model.

QoP parameters from the QoP model are linked with the subset of the facts defined in the model in an explicit way by the linking operator $\mapsto$:

$$QP \longmapsto Z. \quad (14)$$

5.3. Configuration Stage. The next stage is the main phase where all structures proposed in the model for the QoP evaluation of security mechanisms are defined. Among them, one can enumerate definitions of rules, facts order, and the QoP evaluation rules.

5.3.1. Rules Definition. Based on the TLS cryptographic protocol specification [32], we specify the rules which refer to a possible realization of the protocol. In Appendix A one can find the rules defined for the TLS protocol.

5.3.2. Facts Order Definition. In the next step, the facts order must be defined. The order between facts can be defined only for the same security attribute (SA). For the presented example, we evaluate the QoP of security mechanisms in the case of four security attributes: integrity (I), confidentiality (C), authentication (Au), and availability (A). We define the facts order according to the expert knowledge in the field of cryptographic protocols. All the defined fact orders are presented in Appendix B.

5.3.3. QoP Evaluation Rules Definition. The last step in the configuration stage is to define the QoP evaluation rules. For the TLS cryptographic protocol, the evaluation rules refer to the same four security attributes: integrity (I), confidentiality (C), authentication (Au), and availability (A). According to these rules, the QoP evaluation of security mechanism of the TLS protocol is performed. In the literature, defining the influence of specific security mechanisms is based on expert knowledge in the field of cryptology and system security [7, 8]. In our example, we perform the same expert knowledge analysis which refers to the TLS cryptographic protocol. The defined QoP evaluation rules are presented in Appendix C.

5.4. QoP Evaluation Stage. Having finished the configuration stage, one can start the QoP evaluation process. In the presented example we choose six versions of the TLS protocol which are presented in Table 7. These cipher suites are described in detail in the TLS specification [32]. The fifth and sixth versions are modified according to the versions presented in the TLS specification. In the fifth version we analyse a case where a new cryptographic module, compared to the previously defined ones, will be possible. This module is the implementation of HMAC-SHA512 [33]. The sixth version is identical with the first one except that compression is enabled.

The first version is the most popular cipher suite for online banking. Some banks operate their online services using the second TLS protocol version. The third version can be used when authorisation is not required. The fourth version accomplishes the TLS protocol with the strongest set of security parameters. The fifth version analyses the hypothetical scenario when a new implementation of one of the cryptographic modules is possible. The sixth version is identical with the first one, but the data are compressed before encryption.

These six versions are analysed as 6 cases which represent 6 different realizations of the TLS protocol. These versions can be represented as the following set of facts.

Case 1. $C_1 = \{f_1(\text{PK}), f_1(\text{cipher}), f_1(\text{mac}), f_1(\text{key})\}$.

Case 2. $C_2 = \{f_3(\text{cipher}), f_1(\text{key}), f_1(\text{mode}), f_2(\text{mac}), f_1(\text{PK})\}$.

Case 3. $C_3 = \{f_1(\text{cipher}), f_6(\text{PK}), f_1(\text{key}), f_1(\text{mac})\}$.

Case 4. $C_4 = \{f_5(\text{PK}), f_3(\text{cipher}), f_3(\text{key}), f_1(\text{mode}), f_3(\text{mac})\}$.

Case 5. $C_5 = \{f_1(\text{PK}), f_1(\text{cipher}), f_4(\text{mac}), f_1(\text{key})\}$.

Case 6. $C_6 = \{f_1(\text{PK}), f_1(\text{cipher}), f_1(\text{mac}), f_1(\text{key}), f_1(com)\}$.

In Case 5, one can find the fact $f_4(\text{mac})$, which is not defined in Table 5. The example is a situation in which a new protocol implementation will be realized and the QoP evaluation system does not provide analysis for this version. In such a case, this fact will be added later during the detailed analysis of this case.

After defining the facts which describe the analysed cases (C), one has to derive ($\vdash$) other atomic or complex facts (C'): $C \vdash C'$. They can be derived by using the earlier described inference mechanism and the rules defined in Appendix A. On the basis of this knowledge, one can prepare final evaluation of security mechanisms represented as the security attributes. Below we present the evaluation of the six analysed cases.

QoP Evaluation of Case 1.
Consider the following:

$$\begin{gathered}C_1 \vdash C_1'; \\ \begin{aligned} C_1' = \{ & f_1(\text{mac-len}), f_1(k\text{-len}), \neg f_1(IV), \neg f_2(IV), \\ & \neg f_1(\text{bs}), \neg f_2(\text{bs}), f(\text{mac-len}), f(k\text{-len}), \\ & f(\text{PK}), f(\text{cipher}), f(\text{mac}), f(\text{key}), \neg f_2(PK), \\ & \neg f_3(\text{PK}), \neg f_4(\text{PK}), \neg f_5(\text{PK}), \neg f_6(\text{PK}), \\ & \neg f_2(\text{cipher}), \neg f_3(\text{cipher}), \neg f_2(\text{mac}), \neg f_3(\text{mac}), \\ & \neg f_2(\text{key}), \neg f_3(\text{key}), \neg f_4(\text{key}), \neg f_2(\text{mac-len}), \\ & \neg f_3(\text{mac-len}), \neg f_2(k\text{-len}), \neg f_3(k\text{-len})\}. \end{aligned}\end{gathered} \tag{15}$$

Case 1 is denoted as P_1 and is a union of sets C_1 and C_1':

$$P_1 = (C_1 \cup C_1'). \tag{16}$$

The QoP evaluation of the security attributes

$$\begin{aligned} O_1 = \{ & \langle \text{confidentiality}, 2 \rangle, \langle \text{integrity}, 3 \rangle, \\ & \langle \text{availability}, 5 \rangle, \langle \text{authorisation}, 1 \rangle \}. \end{aligned} \tag{17}$$

QoP Evaluation of Case 2.
Consider the following:

$$\begin{gathered}C_2 \vdash C_2'; \\ \begin{aligned} C_2' = \{ & f_2(\text{mac-len}), f_2(k\text{-len}), f_2(IV), \\ & f_2(\text{bs}), f_2(CS), f(\text{cipher}), f(\text{key}), f(\text{mac}), \\ & f(\text{PK}), f(\text{mac-len}), f(k\text{-len}), f(IV), \\ & f(\text{bs}), \neg f_2(\text{PK}), \neg f_3(\text{PK}), \neg f_4(\text{PK}), \\ & \neg f_5(\text{PK}), \neg f_6(\text{PK}), \neg f_2(\text{cipher}), \neg f_1(\text{cipher}), \\ & \neg f_1(\text{mac}), \neg f_3(\text{mac}), \neg f_2(\text{key}), \neg f_3(\text{key}), \\ & \neg f_4(\text{key}), \neg f_1(\text{mac-len}), \neg f_3(\text{mac-len}), \\ & \neg f_1(k\text{-len}), \neg f_3(k\text{-len}), \neg f_1(\text{bs}), \neg f_1(IV)\}. \end{aligned}\end{gathered} \tag{18}$$

Case 2 is denoted as P_2 and is a union of sets C_2 and C_2':

$$P_2 = (C_2 \cup C_2'). \tag{19}$$

The QoP evaluation of the security attributes

$$\begin{aligned} O_2 = \{ & \langle \text{confidentiality}, 8 \rangle, \langle \text{integrity}, 6 \rangle, \\ & \langle \text{availability}, 10 \rangle, \langle \text{authorisation}, 1 \rangle \}. \end{aligned} \tag{20}$$

QoP Evaluation of Case 3.
Consider the following:

$$\begin{gathered}C_3 \vdash C_3'; \\ \begin{aligned} C_3' = \{ & f_1(\text{mac-len}), f_1(k\text{-len}), \neg f_1(IV), \\ & \neg f_2(IV), \neg f_1(\text{bs}), \neg f_2(\text{bs}), f(\text{mac-len}), \\ & f(k\text{-len}), f(\text{PK}), f(\text{cipher}), f(\text{mac}), f(\text{key}), \\ & \neg f_2(\text{PK}), \neg f_3(\text{PK}), \neg f_4(\text{PK}), \neg f_5(\text{PK}), \\ & \neg f_1(\text{PK}), \neg f_2(\text{cipher}), \neg f_3(\text{cipher}), \neg f_2(\text{mac}), \\ & \neg f_3(\text{mac}), \neg f_2(\text{key}), \neg f_3(\text{key}), \neg f_4(\text{key}), \\ & \neg f_2(\text{mac-len}), \neg f_3(\text{mac-len}), \\ & \neg f_2(k\text{-len}), \neg f_3(k\text{-len})\}. \end{aligned}\end{gathered} \tag{21}$$

Case 3 is denoted as P_3 and is a union of sets C_3 and C_3':

$$P_3 = (C_3 \cup C_3'). \tag{22}$$

The QoP evaluation of the security attributes

$$\begin{aligned} O_3 = \{ & \langle \text{confidentiality}, 2 \rangle, \langle \text{integrity}, 3 \rangle, \\ & \langle \text{availability}, 5 \rangle, \langle \text{authorisation}, 0 \rangle \}. \end{aligned} \tag{23}$$

QoP Evaluation of Case 4.

Consider the following:

$$\begin{aligned}
&C_4 \vdash C'_4;\\
C'_4 = \{ & f_3(\text{mac-len}), f_3(k\text{-len}), f_2(IV), f_2(\text{bs}),\\
& f_2(CS), f(PK), f(\text{cipher}), f(\text{key}), f(mode),\\
& f(\text{mac}), f(\text{mac-len}), f(k\text{-len}), f(IV),\\
& f(\text{bs}), \neg f_2(\text{PK}), \neg f_3(\text{PK}), \neg f_4(\text{PK}), \neg f_6(\text{PK}),\\
& \neg f_1(\text{PK}), \neg f_1(\text{cipher}), \neg f_2(\text{cipher}), \neg f_1(\text{mac}),\\
& \neg f_2(\text{mac}), \neg f_1(\text{mac-len}), \neg f_2(\text{mac-len}),\\
& \neg f_1(k\text{-len}), \neg f_2(k\text{-len}), \neg f_1(IV), \neg f_1(\text{bs})\}.
\end{aligned} \tag{24}$$

Case 4 is denoted as P_4 and is a union of sets C_4 and C'_4:

$$P_4 = (C_4 \cup C'_4). \tag{25}$$

The QoP evaluation of the security attributes

$$\begin{aligned}
O_4 = \{ & \langle\text{confidentiality}, 10\rangle, \langle\text{integrity}, 9\rangle,\\
& \langle\text{availability}, 15\rangle, \langle\text{authorisation}, 3\rangle\}.
\end{aligned} \tag{26}$$

QoP Evaluation of Case 5. Case 5 is different from the previous four cases because there is a fact declared (f_4(mac)) which is not in the conditional part of any rule. As we have stated before, a declaration of order between facts is much easier than adding new rules. Based on that, we declare two new orders:

$$\begin{aligned}
& f_4(\text{mac}) >_I f_3(\text{mac}) >_I f_2(\text{mac}) >_I f_1(\text{mac});\\
& f_1(\text{mac}) >_A f_2(\text{mac}) >_A f_3(\text{mac}) >_A f_4(\text{mac}).
\end{aligned} \tag{27}$$

These orders mean that $HMAC - SHA512$ is more than $HMAC - SHA256$ in the context of integrity, but it is also less than $HMAC - SHA256$ in the context of availability.

Adding these orders to the knowledge base results in satisfying conditions of all rules which have f_3(mac) in their conditional parts:

$$\begin{aligned}
&C_5 \vdash C'_5;\\
C'_5 = \{ & f_3(\text{mac-len}), f_3(k\text{-len}), \neg f_1(IV),\\
& \neg f_2(IV), \neg f_1(\text{bs}), \neg f_2(\text{bs}), f(\text{mac-len}),\\
& f(k\text{-len}), f(\text{PK}), f(\text{cipher}), f(\text{mac}), f(\text{key}),\\
& \neg f_2(\text{PK}), \neg f_3(\text{PK}), \neg f_4(\text{PK}), \neg f_1(\text{mac}),\\
& \neg f_5(\text{PK}), \neg f_6(\text{PK}), \neg f_2(\text{cipher}), \neg f_3(\text{cipher}),\\
& \neg f_2(\text{mac}), f_3(\text{mac}), \neg f_2(\text{key}), \neg f_3(\text{key}),\\
& \neg f_4(key), \neg f_2(\text{mac-len}), \neg f_1(\text{mac-len}),\\
& \neg f_2(k\text{-len}), \neg f_1(k\text{-len})\}.
\end{aligned} \tag{28}$$

Case 5 is denoted as P_5 and is a union of sets C_5 and C'_5:

$$P_5 = (C_5 \cup C'_5). \tag{29}$$

The QoP evaluation of the security attributes

$$\begin{aligned}
O_5 = \{ & \langle\text{confidentiality}, 2\rangle, \langle\text{integrity}, 9\rangle,\\
& \langle\text{availability}, 11\rangle, \langle\text{authorisation}, 1\rangle\}.
\end{aligned} \tag{30}$$

QoP Evaluation of Case 6. Case 6 is the same as Case 1 with the difference that the compression is enabled:

$$\begin{aligned}
&C_6 \vdash C'_6;\\
C'_6 = \{ & f_1(\text{mac-len}), f_1(k\text{-len}), \neg f_1(IV),\\
& \neg f_2(IV), \neg f_1(\text{bs}), \neg f_2(\text{bs}), f(\text{mac-len}),\\
& f(k\text{-len}), f(\text{PK}), f(\text{cipher}), f(\text{mac}),\\
& f(key), \neg f_2(\text{PK}), \neg f_3(\text{PK}), \neg f_4(\text{PK}),\\
& \neg f_5(\text{PK}), \neg f_6(\text{PK}), \neg f_2(\text{cipher}), \neg f_3(\text{cipher}),\\
& \neg f_2(\text{mac}), \neg f_3(\text{mac}), \neg f_2(\text{key}), \neg f_3(\text{key}),\\
& \neg f_4(\text{key}), \neg f_2(\text{mac-len}), \neg f_3(\text{mac-len}),\\
& \neg f_2(k\text{-len}), \neg f_3(k\text{-len})\}.
\end{aligned} \tag{31}$$

Case 6 is denoted as P_6 and is a union of sets C_6 and C'_6:

$$P_6 = (C_6 \cup C'_6). \tag{32}$$

This case presents the situation in which we have two conflicting evaluation rules:

$$\begin{aligned}
& f_1(\text{cipher}) \Longrightarrow \text{Inf}^1(A) \quad \text{which we denote as } er_1,\\
& f_1(COM) \wedge f_1(\text{cipher}) \Longrightarrow \text{Inf}^4(A),\\
& \qquad \text{which we denote as } er_{36}.
\end{aligned} \tag{33}$$

The rule er_{36} is subsumed by the rule er_1 because every case which satisfies the rule er_{36} also satisfies the rule er_1. Both of these rules evaluate the same security attribute and both of them, in our case, satisfy the conditions. Based on the previously defined mechanism of recognition of subsuming and conflicting rules, we will treat these rules as conflicting ones and the rule er_{36} will defeat the rule er_1. The QoP evaluation of the security attributes

$$\begin{aligned}
O_6 = \{ & \langle\text{confidentiality}, 2\rangle, \langle\text{integrity}, 3\rangle,\\
& \langle\text{availability}, 8\rangle, \langle\text{authorisation}, 1\rangle\}.
\end{aligned} \tag{34}$$

The results obtained by the QoP evaluation of security mechanisms are presented in Table 8. These results are quantitative.

Table 8: The QoP evaluation of the analysed versions of TLS protocol.

Version	C	I	A	Au
1	2	3	5	1
2	8	6	10	1
3	2	3	5	0
4	10	9	15	3
5	2	9	11	1
6	2	3	8	1

5.4.1. Qualitative Estimation. The results are presented as quantitative estimation of security attributes. During the QoP evaluation of security mechanisms one can introduce qualitative interpretation of the results. That kind of estimation is made for the all security attributes.

In the presented example, we introduce 5 levels of evaluation: very low, low, medium, high, and very high. It is important that the existing correlations between the quantitative and qualitative results have not only a theoretical character but also a real one. A practical character of the qualitative estimation of the security attributes is obtained because the minimal and maximal possible values of the security attributes for a particular version of the analyzed protocol are calculated. The ranges of parameters for the qualitative evaluation are calculated by formula (35).

For the analysed versions of the TLS protocol, the qualitative assessment will be prepared according to the ranges presented in Table 9. These ranges are calculated according to formula (35). On the basis of the calculated ranges, the qualitative assessment of the TLS protocol is obtained. These marks are presented in Table 10:

$$\begin{aligned}
&\text{very low} = \langle Q_{\min}, Q_{\min} + X\rangle\ ;\\
&\text{low} = \langle Q_{\min} + X, Q_{\min} + 2X\rangle\ ;\\
&\text{medium} = \langle Q_{\min} + 2X, Q_{\min} + 3X\rangle\ ;\\
&\text{high} = \langle Q_{\min} + 3X, Q_{\min} + 4X\rangle\ ;\\
&\text{very high} = \left\langle Q_{\min} + 4X, Q_{\min} + 5X\right\rangle\ ,\\
&\text{where}\\
&X = \frac{Q_{\max} - Q_{\min}}{5},
\end{aligned} \tag{35}$$

where

$Q_{\max}$ is the maximum value for the security attribute among all analysed versions of the protocol;

$Q_{\min}$ is the minimum value for the security attribute among all analysed versions of the protocol.

After the QoP evaluation of security mechanisms one can interpret the results. The first and third versions of the TLS protocol are the most efficient ones in the case of CPU performance. This fact is indicated by the availability attribute. This is due to the fact that the applied security mechanisms are the most efficient ones. However, the analysis of confidentiality and integrity indicates that these attributes are accomplished on the lowest level of all analysed versions.

Table 9: The ranges for the qualitative interpretation of QoP evaluation of the analysed versions of TLS protocol.

Mark	C	I	A	Au
Very low	$\langle 2, 3.6\rangle$	$\langle 3, 4.2\rangle$	$\langle 5, 7\rangle$	$\langle 1, 1.4\rangle$
Low	$\langle 3.6, 5.2\rangle$	$\langle 4.2, 5.4\rangle$	$\langle 7, 9\rangle$	$\langle 1.4, 1.8\rangle$
Medium	$\langle 5.2, 6.8\rangle$	$\langle 5.4, 6.6\rangle$	$\langle 9, 11\rangle$	$\langle 1.8, 2.2\rangle$
High	$\langle 6.8, 8.4\rangle$	$\langle 6.6, 7.8\rangle$	$\langle 11, 13\rangle$	$\langle 2.2, 2.6\rangle$
Very high	$\langle 8.4, 10\rangle$	$\langle 7.8, 9\rangle$	$\langle 13, 15\rangle$	$\langle 2.6, 3\rangle$

Table 10: The qualitative interpretation of QoP evaluation of the analysed versions of TLS protocol.

Version	C	I	A	Au
1	Very low	Very low	Very low	Very low
2	High	Medium	Medium	Very low
3	Very low	Very low	Very low	No
4	Very high	Very high	Very high	Very high
5	Very low	Very high	Medium	Very low
6	Very low	Very low	Low	Very low

One of the main functions of the TLS protocol is server authorisation. Versions 1, 3, 5, and 6 guarantee this attribute on the lowest level, but the third version does not guarantee it at all. The fourth protocol version achieves the strongest possible security level. This fact influences the system performance which is indicated by the availability attribute.

In a certain scenario one can use the TLS protocol version which is between the strongest and poorest security levels (*medium*). Then one can use the second version. This version guarantees the confidentiality on the high level and the integrity on the medium level. The authorisation is still guaranteed, but on the very low level. The set of parameters used in the second version allows the decrease of the system performance according to the strongest fourth version to the medium level.

The fifth and sixth versions should be compared with reference to the first version because they are a modification of the first version. In the fifth version, the new cryptographic module which guarantees the integrity on the strongest possible level is introduced. Increasing the level of protection results in a decreased performance, which is indicated by the medium level of the availability attribute. In the sixth version, the compression is enabled which decreases the system performance in comparison to the first version.

5.5. Formal Model Goals Evaluation. At the beginning of the formal model definition we enumerated the goals of our model. In this section, we would like to present the goals achieved by our model.

(1) *Goal 1: Automatic QoP Evaluation of the Not Directly Defined Scenarios.* One of the most important features of the model is the QoP security evaluation based on the not directly

defined scenarios. It means that during the analysis, one can prepare the QoP evaluation of one of the cryptographic protocol versions which is not directly defined. In our model the inference mechanism and rules are defined and owing to that, one can automatically derive the set of facts which describes the analysed version of the protocol. Since our model is based on the analysis of basic mechanisms of security protection and relations between them, the evaluation process does not require an advance preparation of direct scenarios describing all possible configurations of analysed systems. In our model the mechanism which allows for the recognition and resolution of conflicts between evaluation rules is introduced.

(2) *Goal 2: Quality of Protection Evaluation of All Security Mechanisms.* In the proposed model one can evaluate any of the security mechanisms with regard to the quality of protection factor. The security mechanisms, modelled in one of the QoP models, are mapped to the atomic or complex facts. In our model, one can define the facts that any of the security mechanisms can be represented by. The linking stage described in the methodology illustrates the mapping process. Finally, on the basis of the QoP evaluation rules and their order one can prepare the QoP evaluation of security mechanisms.

(3) *Goal 3: Analysis Refers to All Security Attributes*. The security attribute describes system behaviour in terms of information security requirements. That kind of behaviour is changed by the security mechanisms which modify the modelled system. As we mentioned above, in our model one can represent any security mechanism and analyse its influence on any security attribute.

(4) *Goal 4: The model Can Be Used for any QoP Models*. The presented model can be applied to the systems which are modelled by any of the QoP models. This goal is achieved due to the *linking operator* which links the QoP parameters from the QoP model with the subset of facts defined in the model. These objects are mapped in an explicit way.

5.6. Comparison of Our Model with Existing Approaches. In the literature one can find different approaches [4, 7–9] dealing with the quality of protection evaluation of security mechanisms. In Table 11 we compare the model presented in this paper with the existing approaches. These approaches can be characterized by the following main attributes.

> *Quantitative assessment* refers to the quantitative assessment of the estimated quality of protection. All of the presented approaches, except one, allow quantitative assessment of the QoP evaluation of security mechanisms. Petriu et al. [9] discuss performance analysis in terms of the used security level, but the analysis has an qualitative character.
>
> *Formal representation* refers to the representation of the quality of protection evaluation of security mechanisms by mathematical formulae. Among the enumerated approaches only the one presented in this paper has formal representation, while the others are represented by means of an analytical model, without any formal definition of the objects and rules required for formal evaluation.
>
> *Executability* specifies the possibility of the implementation of an automated tool able to perform the evaluation of QoP mechanisms. The tool support is provided for three approaches: Ksiezopolski and Kotulski [4] model is supported by the SPOT tool, Petriu et al. [9] create the UMLsec tool, and the presented model is supported by the SME tool, which can be downloaded from [11].
>
> *Indirect reasoning* reasoning for not directly defined scenarios. All of the presented approaches, except the new model presented in this paper, have one significant limitation. These models can evaluate only these versions which were previously directly defined and described in detail. The presented model provides the method for indirect reasoning.
>
> *Holistic* is the possibility of the evaluation of all security attributes. All presented models, except one, can be used for the evaluation of all security attributes. Only the model presented by Petriu et al. [9] focuses on performance analysis and is related to availability.
>
> *Completeness* is the possibility of the representation of all security mechanisms. This attribute is provided for all models.

6. Conclusions

In the paper we propose a formal model for the quality of protection evaluation of security mechanisms. The presented method has four main features. Firstly, the presented approach allows for the QoP evaluation for the system for which the scenarios of assessment of security mechanisms are not directly defined. Secondly, all security mechanisms can be analysed with regard to the QoP factor. Thirdly, analysis can be performed for all security attributes. Finally, the proposed model can be used for any QoP models.

Our method also has some interesting features which make the evaluation of a system easier. A system can be incrementally developed by adding new knowledge. To avoid conflicts between new and old rules, our model includes the mechanism of recognition and resolution of a specific kind of conflict between evaluation rules.

In the paper we have presented the methodology of our model's preparation. On the basis of this methodology, we have created the model of the TLS cryptographic protocol. We have also performed the QoP evaluation of the sixth selected version of the TLS protocol. The main analysis refers to the quantitative assessment of four security attributes: confidentiality, integrity, availability, and authorisation. Finally, we have introduced the formula for qualitative interpretation of the prepared QoP evaluation.

An additional contribution of the paper is the implementation of the security mechanisms evaluation tool (SMETool) which supports the presented method. The SMETool can be

Table 11: The characterization of the security mechanisms evaluation models.

	Agarwal and Wang [7]	Ksiezopolski and Kotulski [4]	Luo et al. [8]	Petriu et al. [9]	Our model
Quantitative assessment	√	√	√	—	√
Formal representation	—	—	—	—	√
Executability	—	√	—	√	√
Indirect reasoning	—	—	—	—	√
Holistic	√	√	√	—	√
Completeness	√	√	√	√	√

downloaded from the web page of the Quality of Protection Modelling Language Project [11]. The analysed model for the TLS Handshake protocol can be found in the models library in the SMETool.

Appendices

A. The Rules Definition for TLS Cryptographic Protocol

The fact based rules are as follows:

$$f_1(\text{cipher}) \vee f_2(\text{cipher}) \vee f_3(\text{cipher}) \rightarrow f(\text{cipher})$$

$$f_1(\text{bs}) \vee f_2(\text{bs}) \rightarrow f(\text{bs})$$

$$f_1(IV) \vee f_2(IV) \rightarrow f(IV)$$

$$f_1(\text{key}) \vee f_2(\text{key}) \vee f_3(\text{key}) \rightarrow f(\text{key})$$

$$f_1(\text{mac}) \vee f_2(\text{mac}) \vee f_3(\text{mac}) \rightarrow f(\text{mac})$$

$$f_1(mac\text{-len}) \vee f_2(mac\text{-len}) \vee f_3(mac\text{-len}) \rightarrow f(mac\text{-len})$$

$$f_1(k\text{-len}) \vee f_2(k\text{-len}) \vee f_3(k\text{-len}) \rightarrow f(k\text{-len})$$

$$f_1(\text{PK}) \vee f_2(\text{PK}) \vee f_3(\text{PK}) \vee f_4(\text{PK}) \vee f_5(\text{PK}) \vee f_6(\text{PK}) \rightarrow f(\text{PK})$$

$$\sim f(\text{cipher}) \rightarrow \neg f_1(\text{key}) \wedge \neg f_2(\text{key}) \wedge \neg f_3(\text{key}) \wedge \neg f_1(\text{bs}) \wedge \neg f_2(\text{bs}) \wedge \neg f_1(IV) \wedge \neg f_2(IV)$$

$$\sim f(\text{mac}) \rightarrow \neg f(mac\text{-len}) \wedge \neg f(k\text{-len})$$

$$f_1(\text{mac}) \rightarrow f_1(mac\text{-len}) \wedge f_1(k\text{-len})$$

$$f_2(\text{mac}) \rightarrow f_2(mac\text{-len}) \wedge f_2(k\text{-len})$$

$$f_3(\text{mac}) \rightarrow f_3(mac\text{-len}) \wedge f_3(k\text{-len})$$

$$f_1(\text{cipher}) \rightarrow f_1(\text{key}) \wedge \neg f_1(IV) \wedge \neg f_2(IV) \wedge \neg f_1(\text{bs}) \wedge \neg f_2(\text{bs})$$

$$f_2(\text{cipher}) \rightarrow f_2(\text{key}) \wedge f_1(IV) \wedge f_1(\text{bs})$$

$$f_3(\text{cipher}) \rightarrow f_2(IV) \wedge f_2(\text{bs}).$$

The rules which define the exclusive facts are as follows:

$$f_1(\text{cipher}) \rightarrow \neg f_2(\text{cipher}) \wedge \neg f_3(\text{cipher})$$

$$f_2(\text{cipher}) \rightarrow \neg f_1(\text{cipher}) \wedge \neg f_3(\text{cipher})$$

$$f_3(\text{cipher}) \rightarrow \neg f_2(\text{cipher}) \wedge \neg f_1(\text{cipher})$$

$$f_1(\text{bs}) \rightarrow \neg f_2(\text{bs})$$

$$f_2(\text{bs}) \rightarrow \neg f_1(\text{bs})$$

$$f_1(IV) \rightarrow \neg f_2(IV)$$

$$f_2(IV) \rightarrow \neg f_1(IV)$$

$$f_1(\text{key}) \rightarrow \neg f_2(\text{key}) \wedge \neg f_3(\text{key}) \wedge \neg f_4(\text{key})$$

$$f_2(\text{key}) \rightarrow \neg f_1(\text{key}) \wedge \neg f_3(\text{key}) \wedge \neg f_4(\text{key})$$

$$f_3(\text{key}) \rightarrow \neg f_2(\text{key}) \wedge \neg f_1(\text{key}) \wedge \neg f_4(\text{key})$$

$$f_4(\text{key}) \rightarrow \neg f_2(\text{key}) \wedge \neg f_3(\text{key}) \wedge \neg f_1(\text{key})$$

$$f_1(\text{mac}) \rightarrow \neg f_2(\text{mac}) \wedge \neg f_3(\text{mac})$$

$$f_2(\text{mac}) \rightarrow \neg f_1(\text{mac}) \wedge \neg f_3(\text{mac})$$

$$f_3(\text{mac}) \rightarrow \neg f_2(\text{mac}) \wedge \neg f_1(\text{mac})$$

$$f_1(\text{mac-len}) \rightarrow \neg f_2(\text{mac-len}) \wedge \neg f_3(\text{mac-len})$$

$$f_2(\text{mac-len}) \rightarrow \neg f_1(\text{mac-len}) \wedge \neg f_3(\text{mac-len})$$

$$f_3(\text{mac-len}) \rightarrow \neg f_2(\text{mac-len}) \wedge \neg f_1(\text{mac-len})$$

$$f_1(k\text{-len}) \rightarrow \neg f_2(k\text{-len}) \wedge \neg f_3(k\text{-len})$$

$$f_2(k\text{-len}) \rightarrow \neg f_1(k\text{-len}) \wedge \neg f_3(k\text{-len})$$

$$f_3(k\text{-len}) \rightarrow \neg f_2(k\text{-len}) \wedge \neg f_1(k\text{-len})$$

$$f_1(\text{PK}) \rightarrow \neg f_2(\text{PK}) \wedge \neg f_3(\text{PK}) \wedge \neg f_4(\text{PK}) \wedge \neg f_5(\text{PK}) \wedge \neg f_6(\text{PK})$$

$$f_2(\text{PK}) \rightarrow \neg f_1(\text{PK}) \wedge \neg f_3(\text{PK}) \wedge \neg f_4(\text{PK}) \wedge \neg f_5(\text{PK}) \wedge \neg f_6(\text{PK})$$

$$f_3(\text{PK}) \rightarrow \neg f_2(\text{PK}) \wedge \neg f_1(\text{PK}) \wedge \neg f_4(\text{PK}) \wedge \neg f_5(\text{PK}) \wedge \neg f_6(\text{PK})$$

$$f_4(\text{PK}) \rightarrow \neg f_2(\text{PK}) \wedge \neg f_3(\text{PK}) \wedge \neg f_1(\text{PK}) \wedge \neg f_5(\text{PK}) \wedge \neg f_6(\text{PK})$$

$$f_5(\text{PK}) \rightarrow \neg f_2(\text{PK}) \wedge \neg f_3(\text{PK}) \wedge \neg f_4(\text{PK}) \wedge \neg f_1(\text{PK}) \wedge \neg f_6(\text{PK})$$

$$f_6(\text{PK}) \rightarrow \neg f_2(\text{PK}) \wedge \neg f_3(\text{PK}) \wedge \neg f_4(\text{PK}) \wedge \neg f_5(\text{PK}) \wedge \neg f_1(\text{PK}).$$

B. The Facts Order Definition for the TLS Cryptographic Protocol

Orders between facts are as follows:

$$f_3(\text{cipher}) >_C f_2(\text{cipher}) >_C f_1(\text{cipher})$$

$$f_1(\text{cipher}) >_A f_2(\text{cipher}) >_A f_3(\text{cipher})$$

$$f_2(\text{bs}) >_C f_1(\text{bs})$$

$$f_2(IV) >_C f_1(IV)$$

$$f_3(\text{key}) >_C f_2(\text{key}) >_C f_1(\text{key})$$

$$f_1(\text{key}) >_A f_2(\text{key}) >_A f_3(\text{key})$$

$$f_3(\text{mac}) >_I f_2(\text{mac}) >_I f_1(\text{mac})$$

$$f_1(\text{mac}) >_A f_2(\text{mac}) >_A f_3(\text{mac})$$
$$f_3(\text{mac-len}) >_I f_2(\text{mac-len}) >_I f_1(\text{mac-len})$$
$$f_1(\text{mac-len}) >_A f_2(\text{mac-len}) >_A f_3(\text{mac-len})$$
$$f_3(k\text{-len}) >_I f_2(k\text{-len}) >_I f_1(k\text{-len})$$
$$f_1(k\text{-len}) >_A f_2(k\text{-len}) >_A f_3(k\text{-len})$$
$$f_2(\text{PK}) >_{AU} f_1(\text{PK})$$
$$f_3(\text{PK}) >_{AU} f_1(\text{PK})$$
$$f_4(\text{PK}) >_{AU} f_2(\text{PK})$$
$$f_5(\text{PK}) >_{AU} f_2(\text{PK})$$
$$f_4(\text{PK}) >_{AU} f_3(\text{PK})$$
$$f_5(\text{PK}) >_{AU} f_3(\text{PK})$$
$$f_1(\text{PK}) >_A f_2(\text{PK})$$
$$f_1(\text{PK}) >_A f_3(\text{PK})$$
$$f_2(\text{PK}) >_A f_4(\text{PK})$$
$$f_2(\text{PK}) >_A f_5(\text{PK})$$
$$f_3(\text{PK}) >_A f_4(\text{PK})$$
$$f_3(\text{PK}) >_A f_5(\text{PK}).$$

C. The QoP Evaluation Rules Definition for the TLS Cryptographic Protocol

The evaluation rules are as follows:

$$f_1(\text{cipher}) \Rightarrow \text{Inf}^1(A)$$
$$f_2(\text{cipher}) \Rightarrow \text{Inf}^2(A)$$
$$f_3(\text{cipher}) \Rightarrow \text{Inf}^3(A)$$
$$f_1(\text{cipher}) \Rightarrow \text{Inf}^1(C)$$
$$f_2(\text{cipher}) \Rightarrow \text{Inf}^2(C)$$
$$f_3(\text{cipher}) \Rightarrow \text{Inf}^3(C)$$
$$f_1(\text{bs}) \Rightarrow \text{Inf}^1(C)$$
$$f_2(\text{bs}) \Rightarrow \text{Inf}^2(C)$$
$$f_1(IV) \Rightarrow \text{Inf}^1(C)$$
$$f_2(IV) \Rightarrow \text{Inf}^2(C)$$
$$f_1(\text{key}) \Rightarrow \text{Inf}^1(C)$$
$$f_2(\text{key}) \Rightarrow \text{Inf}^2(C)$$
$$f_3(\text{key}) \Rightarrow \text{Inf}^3(C)$$
$$f_1(\text{key}) \Rightarrow \text{Inf}^1(A)$$
$$f_2(\text{key}) \Rightarrow \text{Inf}^2(A)$$
$$f_3(\text{key}) \Rightarrow \text{Inf}^3(A)$$
$$f_1(\text{mac}) \Rightarrow \text{Inf}^1(I)$$
$$f_2(\text{mac}) \Rightarrow \text{Inf}^2(I)$$
$$f_3(\text{mac}) \Rightarrow \text{Inf}^3(I)$$
$$f_1(\text{mac}) \Rightarrow \text{Inf}^1(A)$$
$$f_2(\text{mac}) \Rightarrow \text{Inf}^2(A)$$
$$f_3(\text{mac}) \Rightarrow \text{Inf}^3(A)$$
$$f_1(\text{mac-len}) \Rightarrow \text{Inf}^1(I)$$
$$f_2(\text{mac-len}) \Rightarrow \text{Inf}^2(I)$$
$$f_3(\text{mac-len}) \Rightarrow \text{Inf}^3(I)$$
$$f_1(\text{mac-len}) \Rightarrow \text{Inf}^1(A)$$
$$f_2(\text{mac-len}) \Rightarrow \text{Inf}^2(A)$$
$$f_3(\text{mac-len}) \Rightarrow \text{Inf}^3(A)$$
$$f_1(k\text{-len}) \Rightarrow \text{Inf}^1(I)$$
$$f_2(k\text{-len}) \Rightarrow \text{Inf}^2(I)$$
$$f_3(k\text{-len}) \Rightarrow \text{Inf}^3(I)$$
$$f_1(k\text{-len}) \Rightarrow \text{Inf}^1(A)$$
$$f_2(k\text{-len}) \Rightarrow \text{Inf}^2(A)$$
$$f_3(k\text{-len}) \Rightarrow \text{Inf}^3(A)$$
$$f_1(\text{PK}) \Rightarrow \text{Inf}^1(Au)$$
$$f_2(\text{PK}) \Rightarrow \text{Inf}^2(Au)$$
$$f_3(\text{PK}) \Rightarrow \text{Inf}^2(Au)$$
$$f_4(\text{PK}) \Rightarrow \text{Inf}^3(Au)$$
$$f_5(\text{PK}) \Rightarrow \text{Inf}^3(Au)$$
$$f_1(COM) \land f_1(\text{cipher}) \Rightarrow \text{Inf}^4(A).$$

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The research is partially supported by the Grant "Reconcile: Robust Online Credibility Evaluation of Web Content" from Switzerland through the Swiss Contribution to the enlarged European Union.

References

[1] B. Ksiezopolski, Z. Kotulski, and P. Szalachowski, "Adaptive approach to network security," *Communications in Computer and Information Science*, vol. 39, pp. 233–241, 2009.

[2] B. Ksiezopolski, Z. Kotulski, and P. Szalachowski, "On QoP method for ensuring availability of the goal of cryptographic protocols in the real-time systems," in *Proceedings of the 1st European Teletraffic Seminar*, pp. 195–202, Poznań, Poland, 2011.

[3] J. Jürjens, "Security and compliance in clouds. IT-compliance 2011," in *Proceedings of the 4th Pan-European Conference*, 2011.

[4] B. Ksiezopolski and Z. Kotulski, "Adaptable security mechanism for dynamic environments," *Computers & Security*, vol. 26, no. 3, pp. 246–255, 2007.

[5] B. Księżopolski, D. Rusinek, and A. Wierzbicki, "On the modelling of Kerberos protocol in the quality of protection modelling language (QoP-ML)," *Annales UMCS, Informatica*, vol. 12, pp. 69–81, 2012.

[6] B. Ksiezopolski, D. Rusinek, and A. Wierzbicki, "On the efficiency modelling of cryptographic protocols by means of the Quality of Protection Modelling Language (QoP-ML)," in *Information and Communication Technology*, vol. 7804 of *Lecture Notes in Computer Science*, pp. 261–270, Springer, Berlin, Germany, 2013.

[7] A. K. Agarwal and P. W. Wang, "On the impact of quality of protection in wireless local area networks with IP mobility," *Mobile Networks and Applications*, vol. 12, no. 1, pp. 93–110, 2007.

[8] A. Luo, C. Lin, K. Wang, L. Lei, and C. Liu, "Quality of protection analysis and performance modeling in IP multimedia subsystem," *Computer Communications*, vol. 32, no. 11, pp. 1336–1345, 2009.

[9] D. C. Petriu, C. M. Woodside, D. B. Petriu et al., "Performance analysis of security aspects in UML models," in *Proceedings of the 6th International Workshop on Software and Performance (WOPS '07)*, pp. 91–102, Buenos Aires, Argentina, February 2007.

[10] M. Wazny and G. M. Wojcik, "Shifting spatial attention—numerical model of Posner experiment," *Neurocomputing*, vol. 135, pp. 139–144, 2014.

[11] The official web page of the QoP-ML project, http://qopml.org/.

[12] B. Ksiezopolski, "QoP-ML: quality of protection modelling language for cryptographic protocols," *Computers and Security*, vol. 31, no. 4, pp. 569–596, 2012.

[13] E. LeMay, W. Unkenholz, D. Parks, C. Muehrcke, K. Keefe, and W. H. Sanders, "Adversary-driven state-based system security evaluation," in *Proceedings of the 6th International Workshop on Security Measurements and Metrics (MetriSec '10)*, September 2010.

[14] S. Lindskog, *Modeling and tuning security from a quality of service perspective [Ph.D. thesis]*, Department of Computer Science and Engineering, Chalmers University of Technology, Goteborg, Sweden, 2005.

[15] C. S. Ong, K. Nahrstedt, and W. Yuan, "Quality of protection for mobile applications," in *Proceedings of the IEEE International Conference on Multimedia & Expo*, pp. 137–140, 2003.

[16] P. Schneck and K. Schwan, "Authenticast: an adaptive protocol for high-performance, secure network applications," Tech. Rep. GIT-CC-97-22, 1997.

[17] Y. Sun and A. Kumar, "Quality-of-protection (QoP): a quantitative methodology to grade security services," in *Proceedings of the 28th International Conference on Distributed Computing Systems Workshops (ICDCS '08)*, pp. 394–399, June 2008.

[18] J. Jürjens, *Secure System Development with UML*, Springer, 2007.

[19] T. D. Breaux, A. I. Antón, and E. H. Spafford, "A distributed requirements management framework for legal compliance and accountability," *Computers & Security*, vol. 28, no. 1-2, pp. 8–17, 2009.

[20] M. Theoharidou, P. Kotzanikolaou, and D. Gritzalis, "A multilayer criticality assessment methodology based on interdependencies," *Computers and Security*, vol. 29, no. 6, pp. 643–658, 2010.

[21] ISO/IEC 27001:2005, "Information technology—security techniques—information security management systems—requirements," 2005.

[22] C. Lambrinoudakis, S. Gritzalis, F. Dridi, and G. Pernul, "Security requirements for e-government services: a methodological approach for developing a common PKI-based security policy," *Computers & Security*, vol. 26, no. 16, pp. 1873–1883, 2003.

[23] T. Zurek, "Modelling of a'fortiori reasoning," *Expert Systems with Applications*, vol. 39, no. 12, pp. 10772–10779, 2012.

[24] L. Leszczynski, *Zagadnienia Teorii Stosowania Prawa: Issues of Theory of Application of Law*, Zakamycze, Krakow, Poland, 2001.

[25] W. W. Vasconcelos, A. García-Camino, D. Gaertner, J. A. Rodríguez-Aguilar, and P. Noriega, "Distributed norm management for multi-agent systems," *Expert Systems with Applications*, vol. 39, no. 5, pp. 5990–5999, 2012.

[26] H. Prakken and G. Sartor, "A dialectical model of assessing conflicting arguments in legal reasoning," *Artificial Intelligence and Law*, vol. 4, no. 3-4, pp. 331–368, 1996.

[27] R. A. Kowalski and F. Toni, "Abstract argumentation," *Artificial Intelligence and Law*, vol. 4, no. 3-4, pp. 275–296, 1996.

[28] G. A. W. Vreeswijk, "Abstract argumentation systems," *Artificial Intelligence*, vol. 90, no. 1-2, pp. 225–279, 1997.

[29] J. Hage, *Studies in Legal Logic*, Springer, 2005.

[30] L. W. Torre and Y. H. Tan, "The many faces of defeasibility in defeasible deontic logic," in *Defeasible Deontic Logic*, D. Nute, Ed., pp. 79–121, Springer, 1997.

[31] H. Prakken and G. Vreeswijk, "Logics for defeasible argumentation," in *Handbook of Philosophical Logic*, D. Gabbay, Ed., Kluwer Academic Publisher, 1983.

[32] RFC 5246: The Transport Layer Security (TLS) Protocol v.1.2, 2008.

[33] FIPS 180-3: Secure Hash Standard (SHS).