On the Modelling of the Influence of Access Control Management to the System Security and Performance

Katarzyna Mazur[1], Bogdan Ksiezopolski[1,2] and Adam Wierzbicki[2]
[1]*Institute of Computer Science, Maria Curie-Sklodowska University, pl. M. Curie-Sklodowskiej 5, 20-031 Lublin, Poland*
[2]*Polish-Japanese Academy of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland*

Keywords: Security Modelling, Model-driven Engineering, Model-driven Security, Quality Of Protection, Security Engineering, Access Control Management, RBAC.

Abstract: To facilitate the management of permissions in complex secure systems, the concept of reference models for role-based access control (RBAC) has been proposed. However, among many existing RBAC analyses and implementations, there still exists the lack of the evaluation of its impact on the overall system performance. In this paper, to reduce this deficiency, we introduce an initial approach towards estimation of the influence of the most common access control mechanism on the system efficiency. Modelling RBAC in Quality of Protection Modelling Language (QoP-ML), we analyse a real enterprise business scenario and report obtained results, focusing on time and resource consumption.

1 INTRODUCTION

In large and complex network environments, a proper security management is one of the most challenging issues to be solved. To address this problem, in 1992 David Ferraiolo and Rick Kuhn formalized a role based access control to allow for advanced access control management and reduce the security administration complexity. Over the years role based access control (RBAC) fulfilled authors' promises and became a traditional access control mechanism used widely for implementing several essential security principles, such as least privilege, separation of duties and data abstraction. Modelling RBAC is crucial for many reasons. The main goals for abstracting and implementing the role based access control are reduced costs and improved security. These advantages eventuate primarily due to reduced cost of management of access control and associated directory administration. Related cost savings result from better assets management (including software assets) and simplified audit procedures. Furthermore, RBAC gives excellent scalability and an organisation for security management, providing greater granular permission control by letting administrators assigning general permissions quickly through the use of the preestablished roles. Security managers are able to modify and update existing roles or create specialized ones for users with unusual requirements as needed. Instead of re-assigning privileges to a generous group of users, updating the role automatically updates user's permissions to perform distinct tasks, saving time and resources. Such an approach has a valuable influence on simplifying secure systems administrations, enhancing organizational productivity, system security and integrity. Role based access control maps consistently to the business and organizational structure of an enterprise, allowing for more understandable security policy definition and enforcement. Automation by using RBAC, results in system becoming straightforward to operate and maintain and more secure. Besides the obvious time (and so, indirectly, the energy) savings and security management enhancement, implementing RBAC in the enterprise has also an enormous economic impact. Role based access control decreases system administrator workload, reduces the time for introducing new policy and minimizes the administrative processing time, thereby reducing the cost of security management in large as well as in small enterprises significantly.

In role based access control approach, modelling is an essential technology for managing the interactions that occur across the complex secure systems characterized by the high level of dynamics. It allows not only for the detailed analysis of its individual components, such as roles definitions and permissions assignments, but also the quantification of the authorization level of users allowed to perform vari-

Mazur K., Ksiezopolski B. and Wierzbicki A..
On the Modelling of the Influence of Access Control Management to the System Security and Performance.
DOI: 10.5220/0005378203460354
In *Proceedings of the 17th International Conference on Enterprise Information Systems* (ICEIS-2015), pages 346-354
ISBN: 978-989-758-097-0

ous actions based on the scope of their assigned role. Modelling involves considering a set of interrelated operations within a real, existing system being modelled, that interacts in many different ways, in order to take into account the impact that the design and operational changes on one part of the system have on other parts.

Role based access control has been evaluated in many contexts in a great number of research in a past few years. Among them one can enumerate the economic (O'Connor and Loomis, 2010) or security related studies (rba, 2010). However, to the best of authors' knowledge, the analysis of the influence of the access control management on the system performance seems not to be examined comprehensively enough.

The main contribution of this paper is to model the role based access control structure in Quality of Protection Modelling Language (Ksiezopolski, 2012b). Such an approach allows to evaluate the influence of role and permissions assignment on the system performance. In the article we extended previous studies in the modelling of access management control field to a new context - the performance analysis. Determination of user authorization management efficiency being proposed here, is important because of the proper balancing of the access control management against the security policy, and hence the whole system performance.

2 RELATED WORK

The major goal of the role based access control is to simplify authorization management and review. Having the ability of modelling various access control requirements and facilitating security administration process, RBAC became the object of the study of many researchers. In the literature (Matulevicius et al., 2011), (Sandhu et al., 1996) one can find plenty of RBAC implementations. Preparing RBAC models in SecureUML (Lodderstedt et al., 2002) and UMLsec (Jürjens, 2005) (or any other modelling language available) authors usually focus on its economic or security aspects, omitting the influence of distinct authorization levels on the system performance. However, role based access control has an undeniable impact on performance and should be determined carefully in order to provide the required level of security together with energy efficiency and assurance of the security tradeoffs. To address this issue, many modelling languages and tools have been proposed. Among them one can enumerate UMLsec and SecureUML presented by the researchers in (Matulevicius et al., 2011). Using mentioned approaches, one is able to model and verify secure systems, either pre-existing or those under construction. Nevertheless, introduced solutions focus on developing secure infrastructure or determining system efficiency, rather than examine security and performance concerns at the same time. On one hand, the traditional approach assumes that the implementation of the strongest security mechanisms 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 (Ksiezopolski et al., 2009; Sklavos et al., 2006; Stubblefield et al., 2005). The system performance is especially important in the systems with limited resources such as the wireless sensor networks (Mansour et al., 2014) or the mobile devices. Another example where such analysis should be performed is the cloud architecture. The latest research indicate three main barriers for using cloud computing which are security, performance and availability (Jürjens, 2011). When the strongest security mechanisms are used, the system performance decreases influencing the system availability. This tendency is particularly noticeable in complex and distributed systems. The above statement was as well proved in (B.Ksiezopolski et al., 2011), where the authors, by analysing the performance of the video teleconference connections tunnelled by VPN, evidenced that applied security mechanisms have impact on system performance. They proved that when using strong encryption algorithms, it is impossible to make the video conference of the required quality. Another approach which confirms the above thesis is presented in (uml, 2007). The researchers analyse different security solutions modelled as aspects in UML and examine their performance, and the utilization of both hardware and software resources, using SSL protocol as the example.

The latest results show (Sklavos et al., 2006; Stubblefield et al., 2005; Mansour et al., 2014; Ksiezopolski et al., 2013) that in many cases the best way is to determine the required level of protection and adjust security measures to these security requirements. (Among the means to meet these challenges one can indicate the security metrics (Savola, 2013)). Such approach is achieved by the means of the Quality of Protection models where the security measures are evaluated according to their influence on the system security.

According to the author's knowledge, Quality of Protection Modelling Language (QoP-ML) (Ksiezopolski, 2012b) is the only existing modelling language which satisfies all these requirements simultaneously. It allows for balancing security against the

system's efficiency, performing multilevel analysis of the secure systems and extending the possibility of describing the state of the environment in detail. QoP-ML permits to determine the required quality of protection (QoP) and adjust some of the security measures to these requirements, together with ensuring efficient system performance. This type of profound analysis can be accomplished by the help of the Automated Quality of Protection Analysis tool (AQoPA) (Ksiezopolski, 2012a), which allows for the evaluation of the impact of every single operation defined in the prepared security model in terms of the overall system security. The QoP-ML consists of processes, functions (which change system behaviour), message channels (utilized for communication), variables (used for describing communication channels, processes and functions and storing information about the system or specific process), and QoP metrics (also referred to as security metrics). The process specifies the behaviour, functions represent a single operation or a group of operations, channels outline the environment in which the process is executed. The QoP metrics define the influence of functions and channels on the quality of protection. Complete and comprehensive analysis of the language (the syntax, semantics and algorithms of the QoP-ML) is described in detail in (Ksiezopolski, 2012b).

Since approaches presented in the literature usually speak for an example of a model driven security, in the light of the available development methodologies, QoP-ML excellently fits in a design known as a Model-Driven Engineering. The Model-Driven Engineering (simply known as MDE) is meant to focus on the creation and utilization of the abstract representations of the knowledge that govern a particular domain, rather than on the computing, algorithmic or implementation concepts. Model-Driven Engineering approach is a broader concept than Model-Driven Architecture (MDA), or Model-Driven Security (MDS). MDE adds multiple modelling dimensions and the notion of a software engineering process. The various dimensions and their intersections together with a Domain-Specific Language (DSL) form a powerful framework capable of describing engineering and maintenance processes by defining the order in which models should be produced and how they are transformed into each other. Serving as a domain-specific language, QoP-ML is capable of expressing security models in a formalized, consistent and logical manner.

3 USER ACCESS CONTROL MANAGEMENT ANALYSIS IN TERMS OF THE QOP-ML

In complex secure environments, aside from enhanced security and reduced administration costs, system performance is another important artefact that needs to be carefully evaluated. To improve security management, thereby enhancing the security itself and analyse its impact on the whole system performance, we proposed to prepare the role based access control in Quality of Protection Modelling Language. Modelling a complicated enterprise infrastructure and applying the role based access control is a challenging task. Due to its complexity and the paper's page limitation, complete model can not be presented in detail in the article. However, it can be downloaded from the web page of the QoP-ML project (Ksiezopolski, 2012a). In this section, instead of introducing a complex infrastructure abstraction and considering all the possible operations, we managed to model a simple RBAC usage example in a segment of a real-life business situation, and evaluate its performance with the use of the Automated Quality of Protection Analysis tool (which can be as well downloaded from the web page of the QoP-ML project (Ksiezopolski, 2012a)).

3.1 Scenarios

Modelling RBAC in the Quality of Protection Modelling Language, we assumed the existence of an imaginary, international banking company with corporate headquarters and branches located all over Scandinavian countries. Since the examined enterprise deals with employees assigned miscellaneous responsibilities, and thus distinct permissions and rights to the company's assets, we took under consideration three from the available enterprise roles and determined the influence of different permissions to the overall system performance. To emphasize and prove role's influence on the system's performance, we prepared and analysed the following scenario, which refers to the real business situation and possible role assignment in the actual enterprise environment.

Centralized, role-based security management system acts as the interface between users and the whole system. Given the enterprise network infrastructure in Figure 1, consider having three roles: *security administrator*, *system operator* and *customer* with corresponding security levels: *high*, *medium* and *low*. In our analysis, we assumed the existence of the system which uses a VPN tunnel encryption technology to encrypt the traffic exchanged between all the enterprise locations, since connecting remote offices by

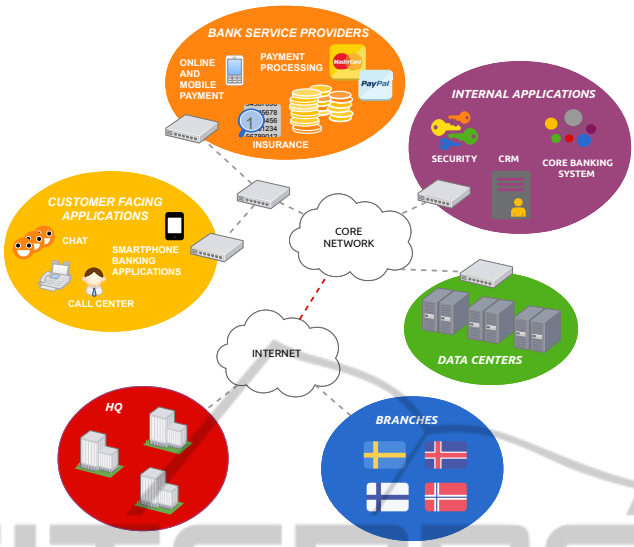

Figure 1: Example enterprise network architecture.

setting up secure VPN tunnels is the preferred way of remote access, especially in the case of e-banking. TLS protocols together with equivalent cryptographic algorithms utilized in our VPN tunnels are summarized in Table 1.

Table 1: TLS protocol versions together with corresponding cryptographic algorithms.

TLS version 1
RC2 +MD5
TLS version 2
3DES/CBC + SHA-1
TLS version 3
AES/CBC 256 + SHA-512

In the examined access control method, users are assigned to specific roles, and permissions are granted to each role based on the user's job requirements. Users can be assigned any number of roles in order to conduct day-to-day tasks. For instance, a single user can have the access to the core banking system (being a *system administrator*), as well as to payment processing systems or customer facing applications (having permissions of the *system operator*). Each role defines permissions that are needed to access different objects. In the proposed scenario, we consider the users assigned *customer* privileges, who have the access to the customer facing applications and use VPN tunnel secured by TLSv1 while making the connection. At the same time, *system operators* dealing with customer facing applications and payment processing exchange traffic with the help of the 3DES-based VPN tunnel. Highest permissions are assigned to the *system administrator* role. Person assigned this privilege set is responsible for system security, customer relationship management system, core banking system and has the control of the *system operators* actions. Prepared scenario is listed in Table 2. Here, in Table 2 by *data size* we understand the amount of the data transferred between the server and the client during the session.

3.2 Role based Access Control Model in QoP-ML

Examining the RBAC approach in Quality of Protection Modelling Language, we prepared the model of the considered enterprise segment, gathered and utilized real hardware security metrics and developed appropriate versions that can be easily mapped to the defined roles.

Within our model we implemented two communicating hosts: a client and a server. The client represents the user (*system administrator*, *system operator*, or *customer*) (in our model referred to as *role3*, *role2* and *role1* respectively) who needs to access one of the enterprise servers through the VPN channel, secured by the mechanisms corresponding to the assigned role. The server process simply manages secure connections, serving incoming requests using suitable TLS versions.

In addition, we prepared three asynchronous communication channels to facilitate the information exchange process. On the client's site, we modelled the main process being responsible for establishing secure connection with the server, and a sub-process capable of generating different types of network traffic based on the role received from the server. Server ab-

Table 2: Scenerio defined for our case study.

Scenario			
RBAC role / RBAC privileges	*customer*	*system operator*	*system administrator*
Application access (single session)	FTP, Web (WWW), Data Center Servers	FTP, Web (WWW), Data Center Servers	FTP, Web (WWW), Data Center Servers
Data size (for each action separately)	10MB each	10MB each	10MB each
Security mechanisms	TLSv1 (see Table 1)	TLSv2 (see Table 1)	TLSv3 (see Table 1)

stracted in Quality of Protection Modelling Language is much alike the client - it also has a main process which sets up the communication parameters, but, opposite to the client, it contains three sub-processes, thereby being able to manage clients with miscellaneous levels of authorization.

Using the QoP-ML's feature called security metrics, we were able to determine modelled system performance on a machines with the actual hardware specifications and thus examine the hardware impact on access control management.

It is worth noting that the flexibility of the utilized modelling language provides the ability to test and simulate proposed roles one-by-one using the versions mechanism.

3.2.1 The Model

In this section we briefly discuss all the elements we prepared to create the role based access control model, and analyse the results we gathered using the Automated Quality of Protection Analysis tool. Scenario (and thus the QoP-ML's security model prepared and used in our case study) can be downloaded from QoP-ML's project web page (Ksiezopolski, 2012a).

Listing 1: QoP-ML's functions.

```
functions {
        fun getRole1();
        fun getRole2();
        fun getRole3();
}
```

On listing 1, we demonstrate functions defined in the quality of protection modelling language, which refer to the roles specified for the example enterprise. Declared operations represent three roles: *system administrator* (first role), *system operator* (second role), *customer* (third role). Remaining functions, referring to the ordinary TLS operations, are omitted in the source code presented below, as being less important for our contribution.

Along with *functions* we declared some *equational rules*. Same as in the case of QoP-ML functions, defined equations are not directly connected to the specified RBAC roles.

Since one needs communication between running processes, it is necessary to define QoP-MLs channels. As channels *ch1* and *ch2* are used to exchange the TLS and actual data traffic, the *rbacCH* is utilized to transfer assigned RBAC role.

Listing 2: Channels used for data transfers.

```
channels {
        channel rbacCH, ch1, ch2(0);
}
```

In our approach, sub-processes express operations that may be performed by users with different RBAC roles assigned. Client and server processes are used to model the TLS handshake operations.

Listing 3: Client's sub-process responsible for accessing one of the available servers.

```
subprocess AccessServer(rbacCH,ch1) {
        in(rbacCH:role);
        if(role == role1) {
                D1 = data()[10MB];
                D1E = enc(D1,K1)[128,RC2,CBC];
                D1MAC = hmac(D1E)[MD5];
                M5 = (D1E,D1MAC);
                out(ch1:M5);
        }
        if(role == role2) {
                D1 = data()[10MB];
                D1E = enc(D1,K1)[56, 3DES, CBC];
                D1MAC = hmac(D1E)[SHA1];
                M5 = (D1E,D1MAC);
                out(ch1:M5);
        }
        if(role == role3) {
                D1 = data()[10MB];
```

```
                D1E = enc(D1,K1)[256, AES, CBC];
                D1MAC = hmac(D1E)[SHA512];
                M5 = (D1E,D1MAC);
                out(ch1:M5);
        }
}
```

After defining processes and sub-processes, one can group them into `host` structures, named *Client* and *Server*, which express the communicating sites in the RBAC model. On the listings below, we present a sub-process of the client who wants to connect to one of the available servers in our scenario, as well as the corresponding server's code, who handles the request, according to the assigned role. Notice, that the server handles distinct roles differently. To present the general concept, we consider only the *system administrator* management behaviour (`HandleRole3`).

Let us examine presented codes briefly. Client's sub-process responsible for connection, firstly reads the RBAC channel (`rbacCH`) to get the assigned role from server. Depending on the obtained permission set, client has the ability to exchange (upload or download) files of various sizes with one of the available servers. Consider the *system administrator* permissions - here, with the help of the `data` function, a data of size `10MB` (defined in the security metrics) is created and assigned to the `D1` variable.

Listing 4: Server's sub-process responsible for handling client's requests.

```
subprocess HandleRole3(rbacCH,ch1) {
        role3 = getRole3();
        out(rbacCH:role3);
        wait()[30];
        in(ch1:Z);
        K1E=Y[0];
        D1E=Z[0];
        D1MAC=Z[1];
        K1=dec(K1E,SKS)[2048,RSA];
        D1EVerif=hmac(D1E)[SHA512];
        if (D1EVerif == D1MAC) {
                D1=dec(D1E,K1)[256, AES, CBC];
        }
        else {
                stop;
        }
}
```

After that, the data is encrypted. In the following step, the `hmac` of the encrypted message is generated and assigned to the `D1MAC` variable. Finally, encrypted message along with its message authentication code is composed in a tuple, and sent through the `ch1` channel to the server. Considering the server's sub-process handling the client's request, one can see that the first instruction uses the `getRole3` QoP-ML's function to generate the permission set for the third RBAC role, and then sends it through the `rbacCH` to the client, using the `out` instruction. After a 30 seconds long waiting time, server uses the `ch1` channel to receive the tuple `Z` (with the help of the `in` operation), consisting of the encrypted message (available in `Z[0]` and assigned to the `D1E` on server's site), together with its corresponding hmac (`Z[1]`). Later on, the server computes its own message authentication code on the received, encrypted message, and checks if it is equal to the obtained one - if so, the server is able to decrypt the message. Otherwise, it ends the communication.

3.2.2 Security Metrics

As mentioned before, system behaviour, which is formally described by the cryptographic protocol, can also be modelled by the proposed QoP-ML. In quality of protection modelling language, the influence of the system protection is represented by the means of functions. During the function declaration, the quality of protection parameters are defined and details about this function are described. These factors do not influence the flow of the protocol, but they are crucial for the quality of protection analysis. During that analysis, the functions QoP parameters are combined with the next structure of QoP-ML, the security metrics. In this structure, one can abstract the functions' time performance, their influence on the security attributes required for the cryptographic protocol or other important factors during the QoP analysis. In the case of the prepared RBAC scenario, we managed to gather actual hardware metrics for all the security mechanisms utilized in proposed TLS versions. To automate metrics generation process, we used Crypto Metrics Tool to measure the performance of the cryptographic mechanisms defined in our case study. Applying statistically validated and free of errors metrics, we were able to evaluate the influence almost as thoroughly as if we use run the simulation on a real hardware.

3.3 Results

In our estimations, we were able to perform the actual analysis for only one client accessing the server in a single session. Remaining results were evaluated to grow linearly, along with the number of incoming session requests.

Consider, for instance, the *customer* role. Here, the user has access to the FTP and WWW servers as well as to the data center resources. A single session between the client and the server can carry the traffic with the maximum size of 30MB. The communication channel is protected by the TLS protocol in version 1 and it takes exactly 8.94 s to perform (Table

3). Nevertheless, having identical conditions, changing only the channel protection type, time extends to 18.72 s (for *system operator*). As *system administrator* is the most secure communication type example (having other conditions equal to *customer* and *system operator* at the same time), it is reasonable to presume that it takes the longest time to accomplish (36.66 s). Such analysis provides serious argumentation to believe that the assigned role can influence the overall system performance.

However, to prove our hypothesis, we estimated the hourly server load of the server being accessed by users with distinct roles. Utilizing results obtained by AQoPA along with those that have been estimated, we evaluated the number of clients (sessions) with different authorization permissions, the server is able to handle within an hour. Our assessment is quite straightforward: knowing that for the existing server, it takes 8.94 seconds to handle user assigned *role1*, and using the simplest possible formula, (1 hour = 3600 seconds, so $3600s / 8.94s \approx 402$) one can say that within an hour server is able to deal with approximately 402 clients (Figure 2).

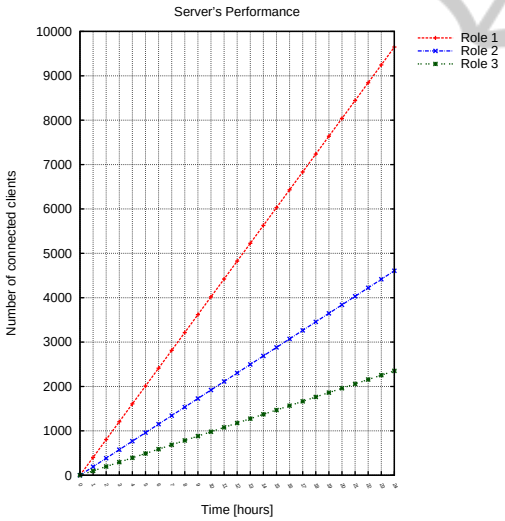

Figure 2: Server's performance in our *scenario*.

Analysing obtained results one can easily notice, that the server is able to handle clients with the first authorization level faster than the same number of clients permitted to perform *system administrator* actions. Gathered results clearly indicate the relationship between the assigned role and consumption of server resources: the longer time the action needs to accomplish, the more server resources are going to be used. A server, which works longer, utilizes more resources, thereby consuming a greater amount of energy.

Table 3: Server's performance results obtained by AQoPA suggest that the assigned role matters if it comes to the system's performance.

Scenario			
RBAC role / Action performed (access)	*customer*	*system operator*	*system administrator*
FTP, Web (WWW), Data Center	2.98s (each)	6.24s (each)	12.22s (each)
Total time (full session)	8.94s	18.72s	36.66s

3.4 Discussion

The time analysis performed for the example, complex IT environment can be a good start for the research on efficient CPU utilization within secure systems. On the basis of the above examination, we estimated the CPU load of server managing users assigned *customer*, *system operator* and *system administrator* privileges. Since the CPU load is usually measured in percentage, we can introduce a simplified CPU utilization formula:

$$U = R/C\,, \tag{1}$$

where:

U - is the CPU utilization, expressed in percentage

R - defines our requirements, the actual busy time of the CPU [seconds]

C - stands for the CPU capacity, the total time spent on analysis [seconds]

The requirements specified in the above formula refer to the time we require from the CPU to perform an action. This time is also known as the *busy* time. CPU capacity can be expressed as the sum of the *busy* and *idle* time (that is, the *total* time available for the CPU). Going simple, one can say that over a 1 minute interval, the CPU can provide a maximum of 60 of its seconds (power). The CPU capacity can then be understood as *busy time* + *idle time* (the time which was used plus the one which was left over). Using the above simplifications, when going multi-core, CPU capacity should be multiplied by the number of the CPU cores ($C = C \cdot cores$). In context of served requests, the presented equation (1) can be further detailed as follows:

$$load[\%] = \frac{time_{session} \cdot users}{time_{total}} \tag{2}$$

where:

$time_{session}$ - refers to the time the single request took [seconds]

users - stands for the number of incoming user connections to be managed

$time_{total}$ - is expressed as $time_{session} \cdot users + time_{idle}$ and represents the total time taken by all the handled connections together with the one which was left over

Calculating CPU load for our scenario, we assumed that *total* analysis time is equal to 3600 seconds (1 hour), and considered 90 users to be handled by the server in each role. Using the (2) equation, we got about 22% of CPU load for the *customer* role, approximately 46 percentage of CPU utilization for clients using TLSv2, and roughly 92% of CPU load for connections protected by AES and SHA-512. Simple analysis proposed above, together with the estimation presented earlier, both prove that the utilized security operations have significant impact on overall system performance. Introduced RBAC model brings the opportunity to evaluate actual differences in chosen security mechanisms and allows selecting the solution which meets our requirements best. Going further, performed research provide serious argumentation to believe that the reduction of the CPU usage, and thus the amount of the utilized energy, entails significant economic profits. Our study showed, that ensuring required security (by switching between the *weakest* and the *strongest* security solutions), it is possible to reduce CPU utilization, and thus, the power consumption and increase cost savings at the same time (dealing with exact number of incoming connections). At first glance, figures presented here may seem irrelevant; however, when put in the context of a large data center environment, they can quickly become very significant.

4 CONCLUSIONS

In the article we used Quality of Protection Modelling Language to prepare the model of example business scenario for the enterprise having role based access control management implemented. We defined two distinct scenarios with dissimilar levels of security, and investigated the performance of the server handling miscellaneous number of users with different RBAC roles assigned. On the basis of the gathered results, we indicated that the user access control management has a meaningful impact on overall system's performance. Our research proved that drawing attention to the system's efficiency while implementing the role based access control policy is crucial from the user access control point of view, usability and security management. Furthermore, the ability of preparing the access control management security model in Quality of Protection Modelling Language, confirmed its extensibility and flexibility with the role based access control functionality.

ACKNOWLEDGEMENTS

This work is supported by Polish National Science Centre grant 2012/05/B/ST6/03364

REFERENCES

(2007). *Performance Analysis of Security Aspects in UML Models*. Proceedings of the 6th International Workshop on Software and Performance.

(2010). *A Comparison of Security Analysis Techniques for RBAC Models*. Proceedings of the 2nd Annual CCWIC.

B.Ksiezopolski, Z.Kotulski, and P.Szalachowski (2011). On qop method for ensuring availability of the goal of cryptographic protocols in the real-time systems. pages 195–202. European Teletraffic Seminar.

Jürjens, J. (2005). *Secure System Development with UML*. Springer.

Jürjens, J. (2011). Security and compliance in clouds. In *Security and Compliance in Clouds*. 4th Pan-European Conference, IT-Compliance.

Ksiezopolski, B. (2012a). The official web page of the qop-ml project.

Ksiezopolski, B. (2012b). Qop-ml: Quality of protection modelling language for cryptographic protocols. *Computers & Security*, 31:569–596.

Ksiezopolski, B., Kotulski, Z., and Szalachowski, P. (2009). Adaptive approach to network security. *CCIS*, 158:233–241.

Ksiezopolski, B., Rusinek, D., and Wierzbicki, A. (2013). On the efficiency modelling of cryptographic protocols by means of the quality of protection modelling language (qop-ml). *LNCS*, 7804:261–270.

Lodderstedt, T., Basin, D., and Doser, J. (2002). Secureuml: A uml-based modeling language for model-driven security. *LNCS*, 2460:426–441.

Mansour, I., Rusinek, D., Chalhoub, G., Lafourcade, P., and Ksiezopolski, B. (2014). Multihop node authentication mechanisms for wireless sensor networks. *LNCS*, 8487:402–418.

Matulevicius, R., Lakk, H., and M.Lepmets (2011). An approach to assess and compare quality of security models. *ComSIS*, 8.

O'Connor, A. and Loomis, R. (2010). Economic analysis of role-based access control. *National Institute of Standards and Technology*.

Sandhu, R., Coyne, E., Feinstein, H., and Youman, C. (1996). Role-based access control models. *IEEE Computer*.

Savola, R. (2013). Quality of security metrics and measurements. *Computers & Security*, 37:78–90.

Sklavos, N., Kitsos, P., Papadopoulos, K., and Koufopavlou, O. (2006). Design, architecture and performance evaluation of the wireless transport layer security. *The Journal of Supercomputing*, 36:33–50.

Stubblefield, A., Rubin, A., and Wallach, D. S. (2005). Managing the performance impact of web security. *Electronic Commerce Research*, 5:99–116.