Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. V (2010), No. 3, pp. 398-409

Role-Based Access Control for the Large Hadron Collider at CERN

I. Yastrebov

Ilia Yastrebov
1. European Organization for Nuclear Research
Switzerland, 1211 Geneva 23, and
2. Joint Institure for Nuclear Research
Russia, 141980 Dubna, 6 Jolio-Curie
E-mail: ilia.yastrebov@cern.ch

Abstract: Large Hadron Collider (LHC) is the largest scientific instrument ever created. It was built with the intention of testing the most extreme conditions of the matter. Taking into account the significant dangers of LHC operations, European Organization for Nuclear Research (CERN) has developed multi-pronged approach for machine safety, including access control system. This system is based on role-based access control (RBAC) concept. It was designed to protect from accidental and unauthorized access to the LHC and injector equipment.
This paper introduces the new model of the role-based access control developed at CERN and gives detailed mathematical description of it. We propose a new technique called dynamic authorization that allows deploying RBAC gradually in the large systems. Moreover, we show how the protection for the very large distributed equipment control system may be implemented in efficient way. This paper also describes motivation of the project, requirements and overview of the main components: authentication and authorization.

Keywords: Software development, role-based access control, information security, equipment protection.

1 Introduction

The Large Hadron Collider was built by the European Organization for Nuclear Research (CERN) with the intention of testing various predictions of high-energy physics, including the existence of the hypothesized Higgs boson and of the large family of new particles predicted by super-symmetry [1]. The LHC is the world's largest and highest-energy particle accelerator. It is contained in a circular tunnel with a circumference of 27 kilometers, at a depth ranging from 50 to 175 meters underground [2]. The LHC uses some of the most powerful dipoles and radio-frequency cavities in existence. The size of the tunnel, magnets, cavities and other essential elements of the machine represent the main constraints that determine the design energy of 7TeV per proton beam [3].

The energy stored in the LHC magnets and beam is enormous, and the potential for crippling machine damage is a serious concern. That's why European Organization for Nuclear Research (CERN) has developed a multi-pronged approach for machine safety [4]:

- Hardware Protection
  - LHC Beam Interlock System
  - Powering Interlock System
- Software Interlock System
- Role-Based Access Control

- Prevents unauthorized access to equipment
- Provide logging to detect errant settings

Role-Based Access Control (RBAC) is an approach to restrict system access to authorized users. Within an organization, roles are created for various job functions. The permissions to perform certain operations are assigned to specific roles. Members of staff (or other system users) are assigned particular roles, and through those role assignments acquire the permissions to perform particular system functions [5]. RBAC is a preventative and therefore inexpensive way to protect the accelerator equipment. It keeps users from making the wrong settings or from logging into the application. Other machine protection systems such as interlocks are reactive and once triggered it is expensive to recover operations.

RBAC is also used to ensure machine stability during a run. Once the equipment is fine tuned and beam is in the machine, an error setting can disrupt operations for hours and lose valuable data. It is important to mention that RBAC is not a security system against hackers; it is designed only to prevent well meaning people from making the wrong setting, and unauthorized users who have no credentials from running the control applications. The last motivation is that RBAC implements logging for each setting protected by access rules. This is crucial during commissioning and debugging. Each setting can be traced and bugs in the sequencer or operations can be caught and corrected [6].

As part of the problem solution a new mathematical model of the role-based access control (hereinafter CERN-RBAC) was proposed. Equipment protection subsystem based on this model has been successfully implemented and deployed at CERN. The subject of this paper is to describe the new concept of CERN-RBAC for distributed equipment control system. The paper also demonstrates the practical implementation of the major system components, and presents the results of the comprehensive testing.

2 Mathematical Model

Role-based access control, as formalized in 1992 by David Ferraiolo and Rick Kuhn [7], has become the predominant model for advanced access control. In 2000, the Ferraiolo-Kuhn model was integrated with the framework of Sandhu et al. [8] to create a unified model for RBAC, published as the NIST RBAC model [9] and adopted as an ANSI/INCITS standard in 2004. Today, most information technology vendors have incorporated RBAC into their product lines, and the technology is finding applications in areas ranging from health care to defense, in addition to the mainstream commerce systems for which it was designed [5]. We propose a new model of CERN-RBAC for the distributed control system. This concept is based on the standard model and preserves the advantages of scalable security administration that RBAC-style models offer. Moreover it significantly extends standard RBAC model according to specific requirements and yet offers the flexibility to specify complex access restrictions based on the dynamic security attributes. The new CERN-RBAC model is quite general and flexible and could be used in many other areas for equipment access control. Below we give a formal mathematical description of the model in terms of sets and relations.

U – A set of users, $\{u_1, u_2, \ldots, u_n\}$. The user is either a human user or a computer program.
R – A set of roles, $\{r_1, r_2, \ldots, r_n\}$. Role is a job function which defines an authority level.
P – A set of permissions, $\{p_1, p_2, \ldots, p_n\}$. Permission (access rule) is an approval of a mode of access to a resource.
UA – User assignment: operation which assigns concrete roles to the users.

$$UA \subseteq U \times R \tag{1}$$

PA – Permission assignment: $R \to 2^P$ – function, defining a set of access rules for each role. This condition must be met at that:

$$\forall p \in P, \exists r \in R : p \in PA(r) \tag{2}$$

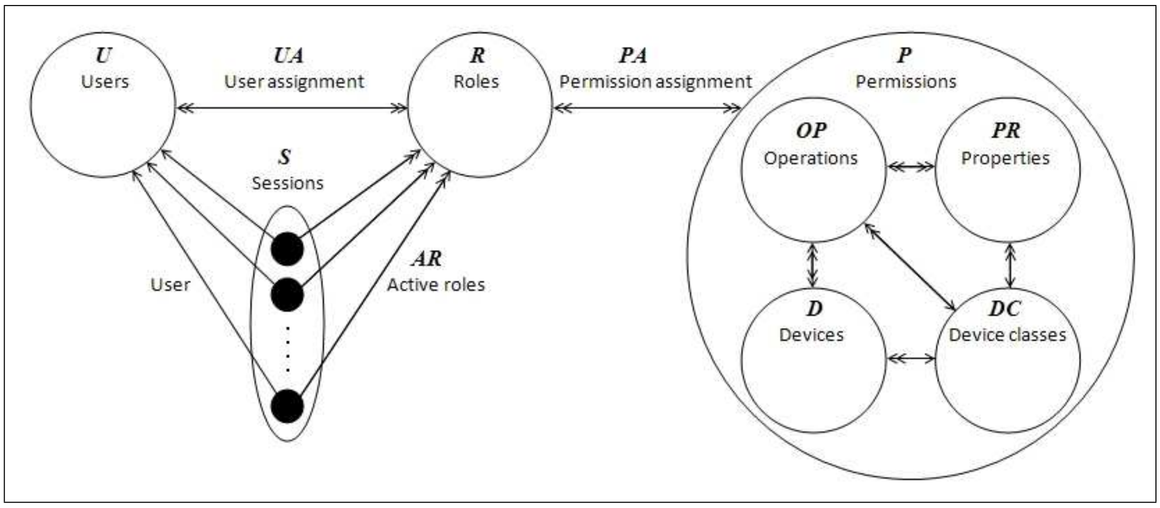

Figure 1: Mathematical model of CERN-RBAC

S – A set of sessions, $\{s_1, s_2, \ldots, s_n\}$. Session (subject) is mapping of one user to possibly many roles. The double-headed arrow from the session to R in Figure 1 indicates that multiple roles are simultaneously activated. The permissions available to the user are the union of permissions from all roles activated in that session. Each session is associated with a single user, as indicated by the single-headed arrow from the session to U in Figure 1. This association remains constant for the life of a session.

$$user(s_i) = S \rightarrow U \tag{3}$$

AR – A set of the subject's active roles (which can change with time). For each subject, the active role is the one that the subject is currently using.

$$AR(s_i) = S \rightarrow 2^R \tag{4}$$

$$AR(s_i) \subseteq \{r \in R \mid (user(s_i), r) \in UA\} \tag{5}$$

Sessions are under the control of individual users. As far the model is concerned, a user can create a session and choose to activate some subset of the user's roles. Roles active in a session can be changed at the user's discretion. The session terminates at the user's initiative. A system may also terminate a session if it is inactive for too long [8].

CERN-RBAC Model

In the previous sections we described the main elements of the RBAC model included in the standard [9]. However the standard model is quite general and abstract to protect the equipment against unauthorized access. The standard concept allows defining permissions only for simple objects. Without further extension it cannot describe more complex interaction. Moreover, the specificity of the equipment requires taking into account its current working mode. That is the authorization algorithm must be dynamic. We propose a new model of the role-based access control for the distributed control system, called CERN-RBAC.

Within this model device is a named entity in the control system. Device can be controlled via its properties. Each property has a name and value. The set of properties specific for the group of homogenous devices forms a device class. The model also defines operations for work with all environments in the system. These are get, set and monitor operations, that allows reading value from device, writing value to device and monitor the device property correspondingly.

OP – A set of operations, $\{get, set, monitor\}$.

PR – A set of device properties, $\{pr_1, pr_2, \ldots, pr_n\}$.
DC – A set of device classes, $\{dc_1, dc_2, \ldots, dc_n\}$.

$$DCA = DC \rightarrow 2^{PR} \tag{6}$$

D – A set of devices (equipments), $\{d_1, d_2, \ldots, d_n\}$. Now we give a function that for every device from the set D defines associated device class:

$$class(d_i) = D \rightarrow DC \tag{7}$$

The properties set for concrete device (APR) are defined as:

$$APR(d_i) = \{pr \in PR \mid (class(d_i), pr) \in DCA\} \tag{8}$$

A set of transactions, which user can execute working within the distributed control system:

$$T = \{(op, pr, d) \mid op \in OP, pr \in PR, d \in D, pr \in APR(d)\} \tag{9}$$

This implies that interaction of user and control system comes to execution of operations from the set OP for the properties of some devices. In terms of proposed model, permission defines for the given role a set of transactions potentially executable by the user owning that role. Function defining a set of transactions for each permission (transaction assignment):

$$TA = P \rightarrow 2^{T} \tag{10}$$

The predicate $exec(s_i, t_j)$ is true if subject can execute transaction at the current time, otherwise it is false. Subject can execute transaction only if it has active roles:

$$\forall s \in S, t \in T, (exec(s,t) \Rightarrow AR(s) \neq 0) \tag{11}$$

Authorization is the function of specifying access rights to resources or services. In our case subject can execute transaction from the set T only if this transaction is authorized for at least one active role of the subject:

$$\forall s \in S, t \in T, (exec(s,t) \Rightarrow t \in TA(PA(AR(s)))) \tag{12}$$

This rule ensures that users can execute only transactions for which they are authorized. Because the conditional is "only if", this rule allows the possibility that additional restrictions may be placed on transaction execution. That is, the rule does not guarantee a transaction to be executable just because it is in the set of transactions potentially executable by the subject's active role [10].

Dynamic Authorization

Equipment functions in different modes. While a device is working in the test mode it's often necessary to allow access for wider range of users than during normal operation. In such cases an expansion of the access rules is not always desirable and appropriate. Firstly, it may loosen up on the system security, and secondly it requires significant administrative costs. We believe the introduction of different working modes of authorization assists to problem solving. In this case the authorization algorithm will take into account not only access rules, but also the current working mode of the equipment.

CERN has a lot of exposed equipments due to the size of the LHC, which contains hundreds of thousands different devices with dozens of properties each. Thus it takes a lot of time to design access rules for all devices. As far as we cannot force equipment specialists to define these rules in a single day, we have to propose the flexible solution to regulating access to non-protected equipments. This will allow us to deploy the access control system step-by-step, without breaking existing infrastructure. It's the crucial requirement for the access control system at CERN.

Dynamic authorization is the algorithm of authorization taking into accounts not only access rules, but also internal state of the authorization subject. Moreover this approach defines default privileges for unprotected environment and non-authenticated users.

CP a set of checking policies considered by dynamic authorization algorithm, {no-check,lenient,strict}. A checking policy is defined at the level of every device and can be changed at runtime. The function mapping each device from the set D to the associated checking policy:

$$policy(d_i) = D \rightarrow CP \tag{13}$$

Now let's introduce the predicate of dynamic authorization as $\psi(s, op, pr, d, policy(d))$, which will be true if subject can execute operation from OP on concrete device property.

$$\forall s \in S, op \in OP, pr \in PR, d \in D : (exec(s,t) \Rightarrow \psi(s, op, pr, d, policy(d))) \tag{14}$$

Now we define a predicate which is true if property is considered to be protected.

$$protected(op \in OP, pr \in PR, d \in D) = 1 \Leftrightarrow (op, pr, d) \in T \tag{15}$$

That is property is protected if there is at least one access rule which restricts access for the given operation. Below we give a description of the dynamic authorization algorithm for each checking policy.

No-check policy grants access for each property without any checking. Typically this policy is used at design stage, when the device interface is not fixed and there are no access rules yet. This policy is also used during testing phase if needed to permit equipment access for some additional users for a short period of time. This mode could be useful for system debugging because or for other activity when it's required to disable CERN-RBAC authorization checks.

$$\psi(no-check) = \begin{cases} 1, protected(pr,d) = 1; \\ 1, protected(pr,d) = 0. \end{cases} \tag{16}$$

Lenient checking policy implements relaxed authorization. For the protected properties algorithm gives access only if corresponding access rule permits so. That is in order to deal with protected properties user must be authenticated in the system. For unprotected properties access is not restricted for any users. Typically this policy is used at the testing stage, when access rules exist only for the most critical settings of the equipment. Some of the devices work in this mode permanently, because sometimes it's desirable to restrict access only to significant settings while keeping others unprotected.

$$\psi(lenient) = \begin{cases} (op, pr, d) \in TA(PA(AR(s))), protected(pr,d) = 1; \\ 1, protected(pr,d) = 0. \end{cases} \tag{17}$$

Strict checking policy implies the most exacting verification. It always requires users to be authenticated in the system; otherwise user's requests will be blocked. Access for the protected properties is granted only if there is an associated access rule. For unprotected properties access is permitted only for reading property value and monitoring operation. Setting new value even for unprotected property is forbidden. This checking policy is the strictest in existence at CERN. Our final goal is to propagate this mode as wide as possible, because it provides best security. All critical equipments are supposed to work in this mode.

$$\psi(strict) = \begin{cases} 0, s = 0; \\ (op, pr, d) \in TA(PA(AR(s))), protected(pr,d) = 1; \\ 1, op \in \{get, monitor\}, protected(pr,d) = 0; \\ 0, op = set, protected(pr,d) = 0. \end{cases} \tag{18}$$

Thereby role-based access control concept with dynamic authorization solves the problem of unprotected properties and legacy applications working without authentication. Introducing this approach we get a system that works dynamically and its behavior can be easily changed at runtime. One of the main advantages of this approach is that it allows deploying CERN-RBAC step-by-step without interruption of the existing software. This requirement is very important for such a huge project like LHC. Our final goal is to protect every single device, but it's impossible to accomplish this job even in one year, taking into account number of environments. We believe that our model of CERN-RBAC could be a good example of the flexible system and could be useful in many other applications for very large distributed systems. Based on the new model of CERN-RBAC we developed and deployed software products. In the following chapters we overview the main technical requirements of the product, and describe implementation of its main components.

3 System Requirements

To distinguish authentication from the closely related term authorization, the shorthand notations A1 (authentication) and A2 (authorization) are occasionally used. In the present paper we list only the most important requirements of the authentication and authorization components of the CERN-RBAC system [11]. Both parts are implemented independently, as two different systems, which do not interact in any way other than passing the users' credentials from A1 to A2.

Authentication requirements:

- Encryption: the credentials used to authenticate the user shall be encrypted when sent over the network.
- Hardware Independence: the method of authentication shall be independent of specialized hardware such as a card reader, finger print reader etc.
- Quick and simple: the method of authentication must be straightforward for the users.
- Authentication Method: authentication shall be done via user name and password from a personal CERN account. The software should allow flexibility for future implementations of Kerberos and/or X.509 certificates.
- Authentication by Location should be implemented as an additional authentication method. This will allow users to be authenticated without providing credentials from a very limited set of trusted machines located in the CERN Control Centre (CCC).
- Role activation mechanism should be implemented, which will allow users to pick up active roles from the set of available roles.

Authorization requirements:

- Subject of authorization: it shall be possible to restrict access i.e. define access privileges for the following operations on each device property: read, monitor, and write.
- Permission Administration: the permissions shall be defined during the design/deployment phase, and authorized administrators shall be able to edit the permissions at any time.
- Logging/tracking: CERN-RBAC shall keep track of all write actions.
- Performance: authorization shall be fast and shall not hinder the performance of the middleware.

4 Authentication

The purpose of the CERN-RBAC authentication system is to verify the digital identity of a principal (which is either a human user or a program). If the authentication succeeds its result is a digitally signed authentication token that is returned to the application [12]. The program can use the token whenever it needs to interact with various parts to the control system. For example, the token can be provided as one of the arguments in a remote call to set a device. Front-ends and the middleware that are receiving such calls will verify the token, thus confirming the identity of the remote party, and can use it as a base for authorization.

The CERN-RBAC authentication token is a short-term uniform substitute of the real credentials. It gets issued by a central service that can reliably verify the user's identity. Various recipients of the tokens can validate them quickly and easily, and use for making authorization decisions [6]. Figure 2 demonstrates the authentication process.
1) Application sends login request to CERN-RBAC.
2) CERN-RBAC checks the user location in the database (for authentication by location).
3) If location is trusted, then the new token immediately created and returned to the user (authentication by location).
4) Otherwise CERN-RBAC sends to the application the login dialog for credentials or certificate. The user types in the name and password or chooses the certificate and sends it to CERN-RBAC.
5) CERN-RBAC verifies the credentials with CERN Account System (NICE).
6) If the check was successful CERN-RBAC retrieves roles from the CERN-RBAC database, generates a new token and signs it with private key to prevent any modifications.
7) CERN-RBAC returns to the application a digitally signed token containing the user roles.

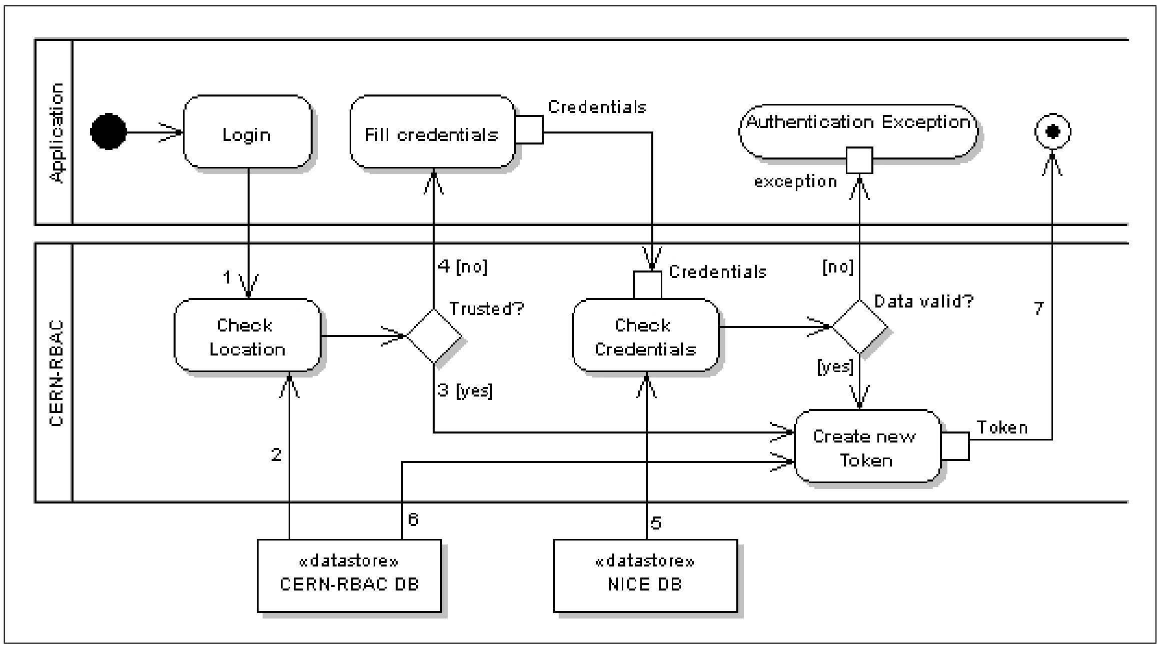

Figure 2: Authentication process

The A1 server is implemented in Java programming language. It receives authentication requests via HTTP from multiple clients, returning back either an authentication token, or an error code. Each request from a user contains its credential in some form. All requests are atomic, so no session information is cached by the server. The SSL/TLS protocol is generally used over HTTP to protect the communication between the two parties, and to authenticate the client's X.509 certificate, if such is provided.

The client side is organized as a library implemented both in C++ and Java, which can be used by other applications or application frameworks. This library provides a function that should be called in order to obtain the authentication token from the server. Java implementation also provides several standard GUI components, such as a login dialog, role picker dialog, and others. Basically, that authentication client-side library can be used in most application without changes. C++ version of the client library is more lightweight because it does not provide any GUI.

Types of Authentication

- By the user name and password. The user names and passwords are checked against the central NICE account database, via a dedicated web service. No user account information is stored in the CERN-RBAC own database.
- By a X.509 certificate. If the user's X.509 certificate is available, it can be applied in the standard client authentication mechanism of TLS/SSL protocol. Then, the certificate information is used to look up the user name in the CERN-RBAC database.
- By the network address (also called Authentication by location). Certain clients can be authenticated by their IP addresses, using a lookup table in the CERN-RBAC database. Normally, the address authentication is permitted only for a very limited number of machines, such as control room consoles.
- By using an existing authentication token. Any existing token can be used to request a new one, providing that the original token is not expired, bears valid signature, and was issued to the same location address. The validity time of the new token will not exceed the validity time of the original one.

5 Authorization

Middleware in the Control System
Controls Middleware (CMW) is a software infrastructure delivered and managed by CERN Beams Department/Controls group. Its goal is to provide a generic way of accessing LHC-era accelerator devices [13]. CMW provides access to devices from application programs in a distributed heterogeneous control system. It allows interconnecting applications and devices implemented in Java or C++, and running under Unix or Windows platforms.

The CMW design reflects the Accelerator Device Model in which devices, named entities in the control system, can be controlled via properties. Each property has a name and a value. The model defines several basic device access methods (get, set, monitor); by invoking these methods, applications can read, write and subscribe to the property values. CMW is based on a client-server model. Accelerator devices are implemented in device servers, and client applications access them using the CMW client API. CMW provides location transparent access to devices: servers can run anywhere in the controls network and devices can be moved from server to server without clients becoming aware of it.

Authorization
CERN-RBAC authorization library is the part of each device server. In most cases, authorization occurs once the application makes a request to get, set, or monitor a property via CMW protocol. This request is made from the application via the CMW client to the CMW server. The token obtained at authentication is passed to the CMW client. There the digital signature is verified, and if valid, the token is sent to the CMW server. If the token is not valid a meaningful exception message is returned to the

sender. The CMW looks up the permission in the access map, and depending on access rights either grants access or blocks the request.

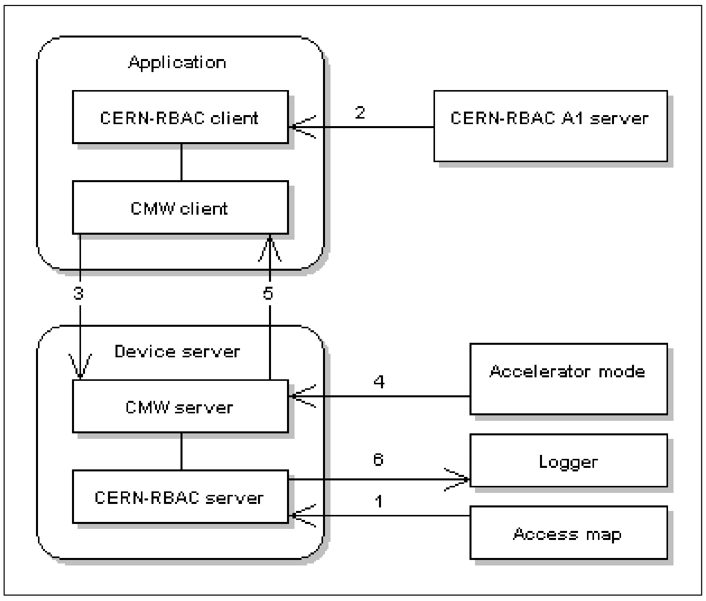

Figure 3: Authorization process

Authorization process:
1) Access map loaded from the local file on device server startup.
2) Client application authenticates on the CERN-RBAC A1 server and obtains a valid token.
3) Token is passed to the CMW client and then to the device server via CMW protocol.
4) CMW server receives the accelerator mode from timing source.
5) Dynamic authorization algorithm verifies user permissions and either allows to execute operation or thrrows an exception.
6) Result of authorization process logged for auditing.

6 Access Rules

Decision whether a particular operation is valid or not is dependent on a set of access rules. They are specified by an equipment specialist for every device class, stored and managed centrally in the Controls database. Every CMW server can read access rules (referred to as access map), relative to device classes it is providing access to, through a tab-separated text file located in the Network File System. This file mirrors the access rules located centrally in the database, and has a digitally signed by the server to prevent any modifications.

Access map is read by CMW server on its start-up. In addition, access map can be reloaded upon a distant call from CMW client. It usually happens when the set of related access rules has been altered by equipment specialist. Access rules are parameterized using all factors relative to the authorization process. More specifically, an equipment specialist has to specify the following fields to define an access rule: device class, property, device name, role for which access is permitted, application, location of the remote client, operational mode of the accelerator, and operation on the property (get, set or monitor). Apart from specific values, an equipment specialist can put a wildcard '*' in any of the fields except device class and operation. This interpreted as all values fit.

Authorization process is performed for each transaction and therefore should work as fast as possible

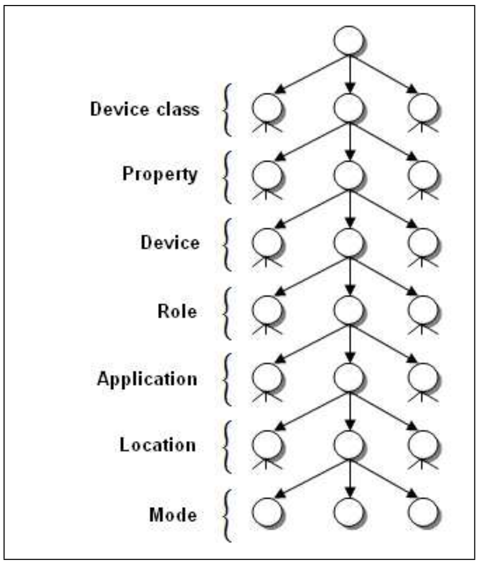

Figure 4: Access Rules structure

and should not slow down the performance of the system significantly. As an authorization time is a concern for the CMW operation, it was decided to represent the access map in the CMW server in the form of trees, a separate one for every operation type. Figure 4 presents the structure of access map tree.

Nodes placed with the same distance from the tree root group authorization argument of one type. Authorization is performed by traversing the tree, appropriate in respect to the operation type, from its root to a leaf, trying at each level to match exact authorization parameter or to find a wildcard '*'. If this succeeds, the access is granted. Such a structuring of the access map allows us to assure authorization with time complexity of O(log n) instructions, where n is the number of access rules [14].

7 Performance and Tests

An important concern with any design is its performance. If CERN-RBAC would double the time to make a setting, nobody would use it. The system also had to scale with a large number of access rules. How does a 2000-rule access map on a front-end impacts the search for the right permission? Another potential performance hit was logging each request on a protected property, and the 3-tier vs. 2-tier performance.

LHC controls can run in 2-tier mode, meaning the application and the client are on one machine and the database and devices are accessed directly on the second machine. This configuration is used when in developer testing. However when the system is in operation the configuration is usually 3-tier. Meaning the client is on a middle tier. A common client is shared by several applications and consolidates requests. In 2-tier, a dedicated client ensures that each request is made from the same application. The CERN-RBAC token can be validated once per session and the credentials can be used for each subsequent request.

In 3-tier, the requests can come from any application at random times. The simplest design is to verify the token for each request. But how does this affect performance? We ran performance tests to answer these questions on a front-end with a 400 MHz. Power PC with Linux, and here are the results:

- The size of the access map has very little effect on performance due to sophisticated and optimized search algorithms. Our test result show a 0.02 ms increase between a 20 rule and 2000 rule access map.

- Logging each request also has very little effect on the performance. Our tests show the difference between having logging on vs. having it off is 0.003 ms.
- 3-tier token verification on every request has a larger impact on performance than the other two concerns. The key size is the most contributing factor. A DSA, 1024 bit key takes 5 ms to validate. A RSA 512 bit key takes 0.150 ms.
- For a 2000 rule access map in a 2-tier configuration the average turn around time or a request is 0.7 ms. In a 3-tier configuration it is 2.7 ms. At this time, this is acceptable according to the requirements.

8 Conclusions and Future Works

The CERN-RBAC approach was successfully implemented based on the proposed model. The system successfully passed many centrally organized tests. Results of the tests prove that the proposed model and design concepts are valid. We also measured the performance of the implemented system and show that the overhead is acceptable. This allows us to assume that the proposed model of the CERN-RBAC could be used in many other areas where access control is needed for large distributed systems.

Currently the RBAC system is released in a production version and used by virtually every equipment device at CERN. Thanks to the algorithm of the dynamic authorization we propagate CERN-RBAC step-by-step, without interruption of the legacy subsystems. In the future we expect to extend functionality of the CERN-RBAC software and the area of its applicability.

Bibliography

[1] CERN: Why the LHC. http://public.web.cern.ch/public/en/LHC/WhyLHC-en.html (2008), Accessed 1 August 2009.

[2] Wikipedia: Large Hadron Collider. http://en.wikipedia.org/wiki/Large_Hadron_Collider (2008), Accessed 1 August 2009.

[3] CERN: What is LHCb. CERN FAQ LHC: the guide. CERN Communication Group. http://cdsmedia.cern.ch/img/CERN-Brochure-2008-001-Eng.pdf (2008). Accessed 9 December 2008.

[4] Wenninger J.: Operational challenges of the LHC. http://irfu.cea.fr/Phocea/file.php?class=std&file=Seminaires/1595/Dapnia-Novc07-partB.ppt. (2007), Accessed 1 August 2009

[5] Wikipedia: Role Based Access Control, http://en.wikipedia.org/wiki/Role_based_access_control (2009), Accessed 1 August 2009

[6] Petrov A., Schumann C., Gysin S.: User Authentication for Role-Based Access Control. *Proceedings of ICALEPCS 2007*

[7] Ferraiolo D.F., Kuhn D.R.: Role Based Access Control. *15th National Computer Security Conference*, Baltimore, USA, (1992)

[8] Sandhu R., Coyne E. J., Feinstein H. L., Youman C. E.: Role-Based Access Control Models. *IEEE Computer* 29 (2): 38-47, (1996)

[9] Sandhu R., Ferraiolo D.F., Kuhn D.R.: The NIST Model for Role Based Access Control: Toward a Unified Standard, *5th ACM Workshop Role-Based Access Control*, 47-63, (2000)

[10] Ferraiolo D.F., Cugini J.A., D. Kuhn D.R.: Role-Based Access Control (RBAC): Features and Motivations. *Proceedings of 11th Annual Computer Security Application Conference*, New Orleans, LA, 241-248, (1995)

[11] Gysin S., Kostro K., Kruk G., Lamont M., Lueders S., Sliwinski W., Charrue P.: Role-Based Access for the Accelerator Control System in the LHC Area - Requirements, EDMS Id 769302, (2006)

[12] Charrue P. et al.: Role Based Access Control for the Accelerator Control System in the LHC Era - Design. EDMS Id 805654, (2007)

[13] Kostro K., Baggiolini V., Calderini F., Chevrier F., Jensen S., Swoboda R., Trofimov N.: Controls Middleware - the New Generation, *EPAC*, Paris, France, p. 2028. (2002)

[14] Kostro K., Gajewski W., Gysin S.: Role-Based Authorization in Equipment Access at CERN. *Proceedings of ICALEPCS*, (2007)

Ilia Yastrebov (b. September 29, 1981) received his M.Sc. in Information Technology (2004). Currently he is a PhD student in Computer Science at Ulyanovsk State University, Russia. He has been working at European Organization for Nuclear Research since 2005 as a software developer for access control. His current research interests include different aspects of Access Control and Distributed Systems. He has 12 papers, 4 conferences participation.