Access Control Design and Implementations in the ATLAS Experiment

Marius Constantin Leahu, *Member, IEEE*, Marc Dobson, and Giuseppe Avolio

***Abstract*—The ATLAS experiment operates with a significant number of hardware and software resources. Their protection against misuse is an essential task to ensure a safe and optimal operation. To achieve this goal, the Role Based Access Control (RBAC) model has been chosen for its scalability, flexibility, ease of administration and usability from the lowest operating system level to the highest software application level. This paper presents the overall design of RBAC implementation in the ATLAS experiment and the enforcement solutions in different areas such as the system administration, control room desktops and the data acquisition software. The users and the roles are centrally managed using a directory service based on Lightweight Directory Access Protocol which is kept in synchronization with the human resources and IT databases.**

***Index Terms*—Access control, databases, roles, security.**

I. Introduction

THE ATLAS experiment comprises a significant number of hardware devices, software applications and human personnel to supervise the experiment operation. Their protection against damages as a result of misuse and their optimized exploitation by avoiding the conflicting accesses to resources are key requirements for the successful running of ATLAS. At the same time the number of users accessing the experiment resources from CERN and external institutes is considerable. Additionally, the users are characterized by a high mobility. Among the access control models implemented nowadays in the computing systems, the Role Based Access Control (RBAC) model fulfills the ATLAS experiment's protection requirements and offers the flexibility to accommodate the high number of users.

The access control issue in the ATLAS Trigger and Data Acquisition (TDAQ) system was addressed first time by an access management component [1] using the RBAC model. This implementation was limited to protection of the applications integrated in the TDAQ system that communicate over a Common Object Request Broker Architecture (CORBA) based inter process communication mechanism.

This paper extends the design of the RBAC model to the scope of the ATLAS experiment with emphasis on the implementation in areas like the system administration, control room desktop and the TDAQ software. It also gives an overview of the users, roles and access policies management using a directory service. The paper focuses on the software infrastructure deployed in the ATLAS online computing system to enable a coherent access control schema in all experiment's subsystems. The access policies (who is allowed to access what, when and from where) will be defined by the ATLAS management [2].

Manuscript received May 9, 2007; revised November 2, 2007.

M. C. Leahu is with PH Department, CERN, CH-1211, Geneva, Switzerland and also with Faculty of Electronics, Telecommunications and Information Technology, Politehnica University of Bucharest, Bucharest 060032, Romania (e-mail: marius.leahu@cern.ch).

M. Dobson and G. Avolio are with PH Department, CERN, CH-1211, Geneva, Switzerland.

Digital Object Identifier 10.1109/TNS.2007.912071

II. Role Based Access Control Overview

The RBAC model takes the access decision for an individual user based on the roles the user has in the organization. The access rights are grouped by role name, and the access to a resource is granted only to users authorized to play the associated role. The NIST RBAC reference model [3] defines four model components: Core RBAC, Hierarchical RBAC, Static Separation of Duty Relations (SSD), and Dynamic Separation of Duty Relations (DSD). The Fig. 1 shows the elements and relations specific to each component.

The Core RBAC defines the minimum set of elements and relations that completely describe a role based access control system. The five basic data elements of the Core RBAC component are:

- USERS: human beings or automated agents;
- ROLES: job functions or job titles which defines an authority level;
- RESOURCE: object which supports a set of possible ACTIONs;
- PERMISSIONS: approvals to perform an action on a given resource.

The key concepts of RBAC are the many-to-many role relations: the user to role assignment (User Assignment) relations and the permission to role assignment (Permission Assignment) relations. The sessions (SESSIONS) are mappings between a user and a subset of roles enabled for the user. It is worth to underline the difference between a user to role assignment relation and a session: the former represents a static assignment of a set of roles to an user (e.g., the "TDAQ Expert" and "TDAQ Shifter" role is assigned to the user when he joins the TDAQ group as an expert and trained to be shifter), while the latest designates subsets of roles already assigned to users and, moreover, enabled (e.g., the user needs only the "TDAQ Shifter" role during his shift, therefore a session with that role enabled is sufficient to accomplish his tasks). The permissions associated to the roles are granted to the users only when the roles assigned to users are enabled. The user session one-to-many relation indicates the session identifiers associated with a user. The session role one-to-many relation gives the roles enabled by a session identifier.

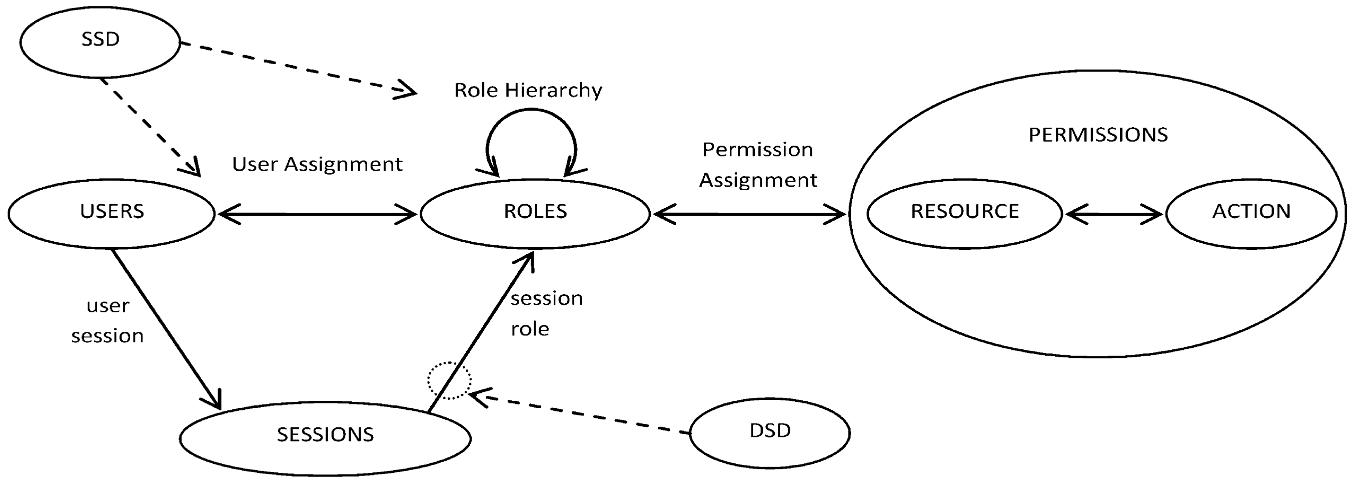

Fig. 1. The RBAC element sets and relations.

The Hierarchical RBAC is the Core RBAC enhanced with the role hierarchy (Role Hierarchy) relations. They are many-to-many relations and define inheritance relations among roles that is, role A inherits role B if all permissions granted to role B are also granted to role A.

The constraints on the relations between elements take the form of Static Separation of Duty (SSD) relations and Dynamic Separation of Duty (DSD) relations. The SSD relation specifies constraints on the assignment of users to roles. Thus if a user is authorized as a member of one role, the user is prohibited from being a member of a second role. This constraint is inherited also within a role hierarchy. For example, if the ATLAS operation policy requires that the security officer and the shift leader must be two different persons, the role ATLAS Security Officer and ATLAS Shift Leader are in a SSD relation. The DSD relation puts restriction on the roles that can be enabled within a user's session. The role TDAQ Database Administrator who is able to modify the TDAQ software configuration database is in a DSD relation with the role TDAQ Shifter who is allowed to start the TDAQ software because the configuration database must not be changed during the TDAQ software execution.

III. System Design

The ATLAS experiment operational model [4] defines activity areas with their tasks and responsibilities for various systems. The organizational structure resulting from this operational model is naturally reflected by the roles associated to systems. Each system in turn is organized internally in subsystems and the best mapping of this characteristic is the role hierarchy. Therefore, the Hierarchical RBAC model is the best approach for an access control schema in the ATLAS experiment. At the same time, the systems considered critical for the experiment operation may require a higher protection level by enforcing the SSD or DSD constraints.

In summary, the ATLAS access control system implements the Hierarchical RBAC model and allows for extensions with SSD or DSD constraints.

A. *Access Control System Architecture*

The scope of the ATLAS access control is the ATLAS online computing system [5] where most of the hardware and software resources are located or are controlled from. This wide scope requires a design for the access control system that allows for a centralized management of the access policies and, at the same time, permits implementations from the lowest operating system level to the highest software application level.

Fig. 2 provides an overview of the ATLAS access control software entities and their relations to meet these requirements. The following paragraphs detail the main software entities in the ATLAS online computing system involved in the implementation of the RBAC.

The *databases* are the coordination points for the access control within the experiment context. All the RBAC elements and relations are stored in there: the users, the roles, the permissions, the roles hierarchies, the user assignments to role, and the permission assignment to role. The databases are the single point of storage for the RBAC constituents. There are multiple advantages of this approach: less data repositories to secure, less redundancy mechanisms to implement, and better control over the coherence of the access control policies among all the systems. The need for more than one database is detailed later in this paper.

The access control policies are defined in the databases as RBAC relations between users, roles and permissions. The enforcement of these access control policies is the responsibility of the software entities interested in the protection of their functionality and the data they manipulate.

The *operating system* running on the cluster nodes is the software entity with the largest spectrum of functionalities. The cluster nodes play various roles in the online computing system from the service providers (e.g., file servers, boot servers, application gateways, network services) to processing nodes with functions in the data acquisition. The users' access to each of the cluster nodes is restricted with the operating system native access control mechanisms and, in addition, taking into account the role of the user in the system. The mechanisms are detailed later in the system administration enforcement example.

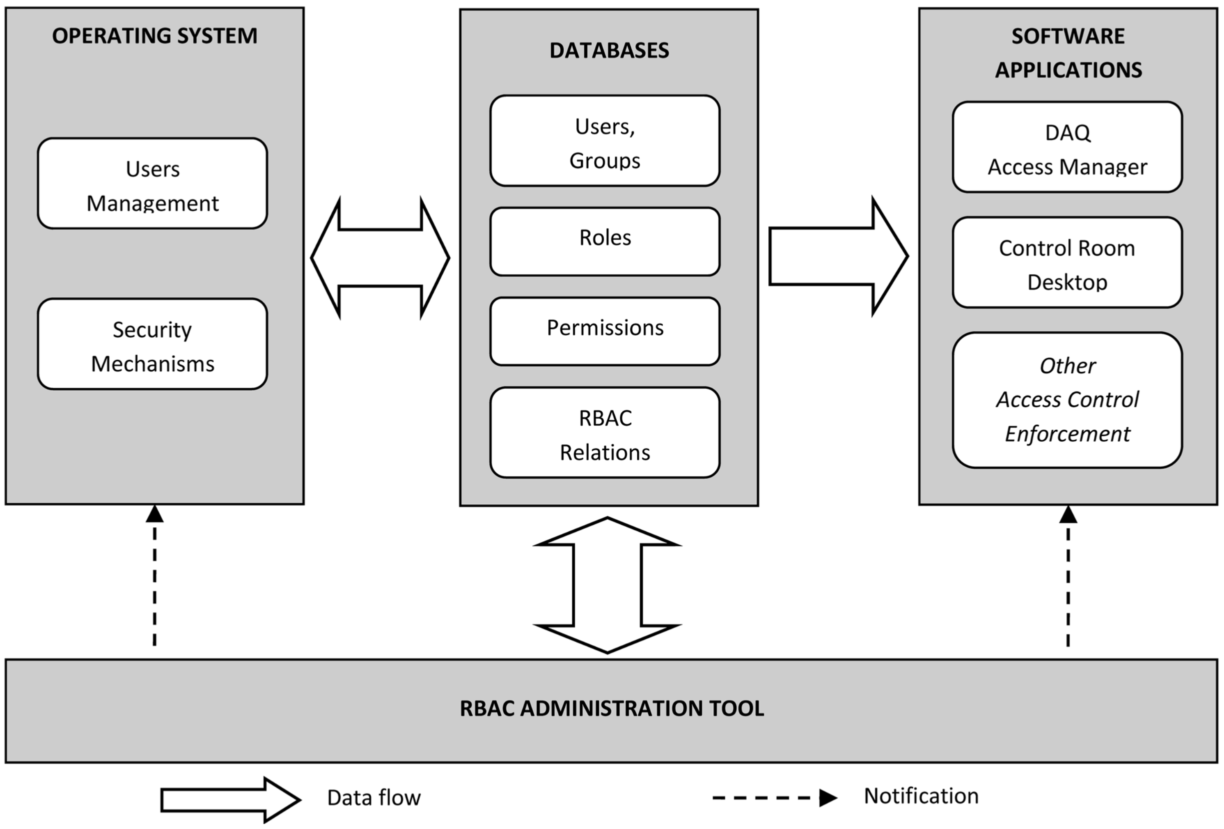

Fig. 2. Access control software entities in the ATLAS online computing system.

The *software applications* executed in the operating system environment may be able to control directly hardware devices in the ATLAS experiment infrastructure or to access data sources with configuration parameters for hardware and software systems. These types of software entities are also subject to access control restrictions and they are responsible for enforcing the policy in their working area.

While the databases are the passive entities in the access control implementation, the operating system and the software applications are active entities which enforce the access control policies.

The last software entity is the *RBAC Administration Tool* which has the most complex set of functionalities and is presented in detail in the following section.

IV. RBAC Administration Tool

This software entity is responsible for the management of most of the RBAC elements sets and relations. The functionalities of the RBAC Administration Tool are split over three categories corresponding to the following components: Administrative Component, Users Sessions Component, and Review Component. There are two categories of users for this tool: the administrators and the shifters. The administrative component is controlled by the administrators, the users sessions component by the shifters and the review component by both user categories. The following paragraphs detail the RBAC Administration Tool components enumerated above.

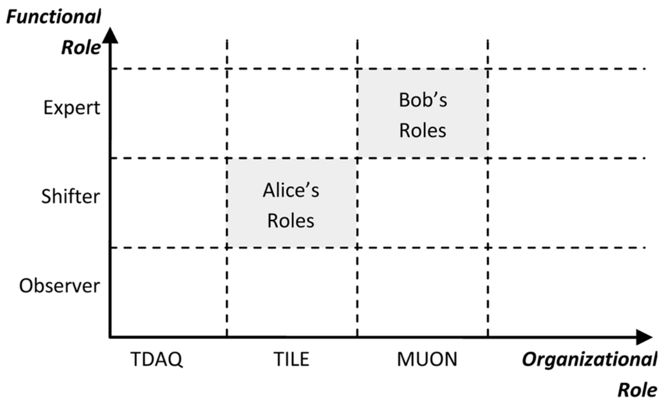

Fig. 3. The organizational and functional roles.

A. Administrative Component

This component is in charge of the creation and maintenance of the RBAC elements sets and relations: create and delete roles, define resources and actions, define permissions by associating actions to resources, build role hierarchies, assign/revoke roles to/from users, assign/revoke permissions to/from roles, and allow to define SSD or DSD relations. The user accounts creation and deletion operations are not part of this component's functionalities and are detailed later in the section "Databases and Synchronization".

The Fig. 3 represents the two types of roles: the organizational roles and the functional roles. The organizational roles correspond to the ATLAS administrative groups or projects (e.g., subdetector group such as Hadronic Calorimeter (TILE), Electromagnetic Calorimeter (LAR), Muon, or TDAQ). The functional roles are associated to the expertise levels within an administrative group or project (e.g., administrator, expert, shifter, or observer). The set of roles assigned to a user comprises pairs of both types of roles in order to better describe the user's abilities and permissions in the ATLAS system. For example, the user Alice is a member of the TILE group and has been trained to be a shifter, so she is assigned to the role TILE-Shifter (inherits the organizational role TILE and the functional role shifter). The user Bob member of the MUON group has the highest level of expertise, so he is entitled with the MUON-Expert role (inherits the roles MUON and shifter).

The role hierarchies incorporate the knowledge about the administrative group and project hierarchies in the ATLAS experiment. The hierarchy granularity can be increased by defining roles per tasks within the same group or project. The hierarchies are defined for the functional roles as well: the shifter includes the observer's privileges, and the expert includes the shifter's privileges for example. However, in order to keep the administrative task as intuitive as possible, the roles hierarchies are limited to tree or inverted tree structures.

The resources types and the actions are predefined for each software entity designed to incorporate an access control mechanism. Therefore, the Administrative Component defines permissions in terms of what action is allowed on what resource value. For example, the predefined resource type "process manager" with the attribute "process name" has associated the predefined action types "start" and "terminate" with the attribute "time". A permission in this case could be: Allow action type "start" with value "12:00" for attribute "time" to be performed on the resource type "process manager" with value "kdestart".

An important and difficult task of the Administrative Component is to check the coherence of the overall access control policies. The lack of correlation between permission definitions, users to role assignments and SSD/DSD specifications can hide or lock system resources or actions on resources. An example of two uncorrelated access rules are the following: allow the role A to start TDAQ processes, and allow the role A to log in only to the public computers (where the TDAQ binaries are not accessible). In this case, the TDAQ process resource is hidden for the role A.

B. *Users Sessions Component*

The users in the ATLAS experiment have at most one session active at a time. The session creation and destruction is the responsibility of the Users Sessions Component. When a user session is created, a subset of roles is enabled from the set of roles assigned to the user. The enabling operation checks in addition the validity of the enabled subset of roles against the DSD relations if any were defined. The user session is visible in all the software entities with the access control implementations, which are responsible for policy enforcement with the subset of roles enabled in the session.

C. *Review Component*

The administrator and shifter users may want to view the current status of the RBAC elements and relations. This goal is accomplished by the Review Component. It is able to display the permissions, the roles, the roles inheritance relationships, the SSD/DSD relations, the user-to-role assignments, and the permission-to-role assignments.

The Administrative Component and the Users Sessions Component can change the RBAC elements and relations during the ATLAS experiment operation, and the new access policies must be enforced immediately everywhere. The access control implementations may cache some RBAC specific data to minimize the performance impact over the system they protect, so the cache is invalidated each time the RBAC Administration Tool notifies them about a change in the RBAC elements or relations.

The multitude of functions the RBAC Administration Tool is able to perform and their consequences on the good operation of the ATLAS experiment, makes from this tool a good candidate for an access control implementation. Indeed, the tool should protect itself with access control policies allowing only the administrators and shifters users to control it.

V. Access Control Enforcement

This section gives an insight on how the RBAC is enforced in the System Administration area, the Control Room Desktop implementations, and the TDAQ software.

A. *System Administration*

The access control in the System Administration concerns the protection of the software infrastructure in the ATLAS experiment and the operating system running on each online computing node. Since more than 90% of the machines are running a Linux operating system (based on Scientific Linux CERN distribution), the following discussion focuses on that OS.

The software infrastructure [5] provides generic services like the network services [Domain Name System (DNS), Network Time Protocol (NTP), Dynamic Host Configuration Protocol (DHCP)], the network booting service for the online computing nodes (network booted for easy administration), the users information, and the users home directories.

The first protection level for the ATLAS experimental area is provided by the Application Gateways which separate the ATLAS Control Network from the rest of the CERN network. The users outside the experimental area and wishing to login to machines in the ATLAS Control Network have to first login to the Application Gateways and then hop to the desired machine. The Application Gateways are the single access point to the experimental area and it is therefore the best place to implement access control. The restrictions at this level are simple and check for example the user account validity as an ATLAS account. The restrictions can be more drastic depending on the experiment status (e.g., while the experiment is running, only the users with expert roles are allowed to login).

Once the users have hoped through the Application Gateway, they are free to attempt to login to any machine in the experimental area (the online computing nodes, the file servers etc). However, each machine is protected with the Linux native security mechanisms extended to take into account the user's roles defined by ATLAS policy and the machine's functionality.

The computers functionalities in the ATLAS experimental area vary from public nodes where the users can display web

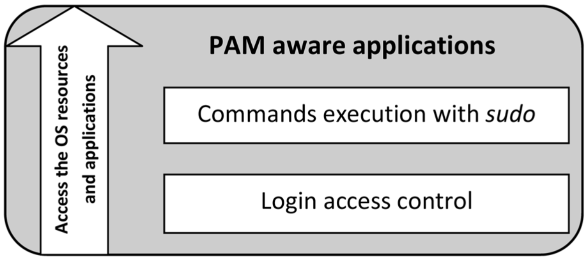
PAM aware applications
Commands execution with sudo
Login access control
Access the OS resources and applications

Fig. 4. Layered access control on Linux nodes.

pages of the safety information to the control room nodes where the detectors operational parameters can be changed. While the security level should be minimum for the first case (the users must be able to view the safety instructions as quickly as possible without any role checking, but only the security officers must be able to edit and update these instructions), the second case needs the maximum security level (the user at the control room desktop console must have all the necessary roles enabled to be able to change the detector running conditions). This variety of computer functionalities requires a solution to configure the access control granularity for each individual node. This goal can be achieved by adding more access control filters with finer restrictions each time the security level needs to be increased for a node, so that the node's access control is in the end a stack of access control filters.

The layered approach for access control is implemented on the Linux machines in the ATLAS experiment with the Pluggable Authentication Module (PAM) [6] as represented in the Fig. 4. The RBAC check is integrated in the Linux security mechanism as a PAM module which is used by any PAM aware application to enforce an access policy. The local and remote login (e.g., Secure Shell), and the *sudo* tools are a few examples of tools using the PAM.

The file server machines allow only the users with administrator roles to login and the execution of some commands (e.g., *reboot*, *fdisk*) is restricted to the administrator or experts. The online computing nodes can restrict the login access only to the users from the detector group the nodes belong to. These two examples are possible use cases of the layered access control on the Linux nodes.

The few machines running the Windows OS are used by the ATLAS Detector Control Systems (DCS) where they run the PVSS SCADA software [7]. The access control is then implemented at the application level by the PVSS Access Control tool [8] extended on top of JCOP framework [9]. This extension was designed to meet CERN's requirements on access control [10]. This development has enabled the use of the RBAC specific data from the databases presented in the previous section to enforce the ATLAS access policies.

B. Control Room Desktop

The Control Room Desktop (CRD) is the GUI environment available on the desktop machines in the ATLAS Control Room. It is based on the Linux K Desktop Environment and exploits its KIOSK [11] configuration mode. The KIOSK framework provides a set of features to easily and powerfully define and restrict the capabilities of a KDE environment based on the user's credentials. It permits the construction of a controlled environment by customizing and locking almost all the desktop functionalities. The restrictions range from disabling the background wallpaper customization to disabling the user log out button or the possibility to access a command shell.

When the user logs in to a control room machine, the CRD's access control retrieves the RBAC data from the databases and decides what KIOSK profile to load for the user. The GUI environment offered to the user is generated from the KIOSK profile so the user has access only to the functions and applications specific to his current roles.

C. TDAQ Access Manager

The management and control aspects of the TDAQ software are covered by the TDAQ Control and Configuration [12] software components. The Access Manager component is responsible for the access control implementation in the TDAQ software. The component has a client—server architecture where the client sends authorization requests to the server, and the server processes the requests and sends back the responses to the client.

The clients are other TDAQ Controls software components that need to protect their functions based on an access control policy. A few examples are the Process Manager, Run Control, or IGUI [12]. The decision task is complex and requires communications with the databases holding the RBAC specific data. The optimal solution is to implement the decision algorithm in a separate software component that is optimized to process and deliver access control decisions as fast as possible. The Access Manager server accomplishes this task. The clients are calling functions provided by the Access Manager Client Application Programming Interface (API) to ask for authorization and take the appropriate action according to the response received from the server.

The Access Manager server is designed as an highly scalable server architecture based on the reactor pattern to handle hundreds of client requests in parallel. It retrieves the RBAC data from the databases, listens for authorization requests from clients, processes the requests with the access policies taking into account the user's roles enabled at that time, and communicates the response to the client.

The Access Manager implementation contains the server part developed in Java and the client API implemented in Java and C++. Currently there are three TDAQ Controls components that successfully use the Access Manager Client API.

The Access Manager tests on the test bed cluster validated the designed functionality and revealed a maximum server processing rate of $\sim$500 authorization requests per second on hardware configurations similar to today's desktop computers. The next suite of tests will be run on the experiment preseries cluster with more powerful hardware configuration where a significant increase in performance is expected.

VI. Databases and Synchronization

The CERN users and implicitly the ATLAS experiment users are administrated by the Human Resources (HR) department in a centralized HR database. The department is responsible for keeping the users information up to date, so the HR database is the reference database for the CERN user computing accounts database. The CERN IT department administrates the CERN user computing accounts and keeps them synchronized with the HR database. At the same time, one of the ATLAS experiment requirements is to be able to run and take data for up to 24 hours while disconnected from the CERN IT department. Consequently, the direct use of the CERN user computing accounts database by the ATLAS computing system would make the experiment impossible to run. The solution is to mirror the ATLAS user accounts from the CERN user computing accounts database in a database located in the experimental area. The ATLAS user accounts database must be synchronized with the CERN database and this imposes constraints on the technical solutions for the ATLAS user database.

The technical solution for the ATLAS databases with RBAC specific data has to allow for easy interrogation from all the access control software entities presented in the previous sections. The user information needed by the Linux nodes in the ATLAS experiment is stored in the ATLAS user accounts database. Therefore the user management solution for Linux nodes should be able to query a central database for user information. The solutions for centralized user management on the Linux OS use a directory service that stores the user information, and the most popular implementations are based on the Sun's Network Information Service (formerly Yellow Pages) or the Lightweight Directory Access Protocol (LDAP).

The LDAP solution has been chosen because it offers the advantage of storing more information besides the user information (e.g., centralized configurations for Linux utilities such as *autofs* or *sudo*). Also, the LDAP interrogation from the software applications entities is facilitated by the LDAP API's available for many programming languages.

The current directory service used in the ATLAS online computing system is the OpenLDAP [13] implementation included in the Scientific Linux CERN distribution. There is one LDAP server instance that hosts almost 300 users (to increase later up to 1000), Linux tools configurations (*autofs*, *sudo*, *samba*, *Network Information Service (NIS) netgroups* information for system access control), and RBAC elements and relations (roles, role hierarchies, roles assignment to users). The number of computers accessing the LDAP server is approximately 500 and this will increase to almost 2500 in the final ATLAS online computing system.

VII. Conclusion

The ATLAS access control design based on the RBAC model fulfils the ATLAS requirements in terms of security and operation autonomy. The detailed technical requirements for the access control software entities have been finalized and their implementation and testing progress with the ATLAS experiment installation.

The cornerstone of the RBAC deployment in the ATLAS experiment is the definition of the roles elements, role hierarchies and most important the permissions and the assignment of permissions to roles.

Currently there is a prototype RBAC setup used by all the access control enforcement examples detailed in this paper. The prototype comprises four functional roles (observer, shifter, expert and administrator) and two top organizational roles (TDAQ, DCS) with subroles for each subdetector. The Linux server nodes restrict the login access to the system administrator roles and *sudo* configurations are centralized in the OpenLDAP server. The PVSS Access Control module is configured to use the prototype RBAC setup, and the Control Room Desktop contains KIOSK profiles for the prototype roles. The TDAQ Access Manager tests have been run with access policies defined for each TDAQ component implementing access control and each TDAQ functional role (e.g., TDAQ observer, TDAQ shifter).

The mechanisms to synchronize the ATLAS user database with the CERN databases are being finalized and the scalability tests are ongoing. We expect to find a performance limitation with only one LDAP server, so a research project has been started to identify solutions to improve the directory service in the ATLAS system in terms of scalability and robustness.

Acknowledgment

The first author, M. Leahu, would like to thank G. Lehmann Miotto and G. Unel for their valuable reviews.

References

[1] J. E. Sloper, M. Leahu, M. Dobson, and G. Lehmann, "Access management in the ATLAS TDAQ," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 3, pp. 986–989, Jun. 2006.
[2] G. Lehmann and S. Stapnes, "Policy for access to the ATLAS online systems," Jun. 2006 [Online]. Available: http://acr.web.cern.ch/acr/onlinesystemaccess-v1.1.pdf
[3] D. Ferraiolo *et al.*, "Proposed NIST standard for role-based access control," *ACM Trans. Inf. Syst. Security*, vol. 4, no. 3, pp. 224–274, Aug. 2001.
[4] ATLAS Operational Model Diagram [Online]. Available: https://twiki.cern.ch/twiki/bin/view/Main/OperationModelDiagram
[5] M. Dobson *et al.*, "The architecture and administration of the ATLAS online computing system," presented at the CHEP, Mumbai, India, 2006.
[6] A. G. Morgan, "Pluggable authentication modules for Linux," 2005 [Online]. Available: http://www.kernel.org/pub/linux/libs/pam
[7] ETM Professional Control GmbH, Eisenstadt, Austria, "SCADA system (supervisory control & data acquisition)," 2007 [Online]. Available: http://www.pvss.com
[8] CERN, Geneva, Switzerland, "JCOP framework access control," 2007 [Online]. Available: http://itcobe.web.cern.ch/itcobe/PrePublication/Projects/Framework/Download/Components/AccessControl
[9] O. Holme *et al.*, "The JCOP framework," presented at the 10th Int. Conf. Accelerator Large Experimental Phys. Control Syst. (ICALEPCS), CICG, Geneva, Switzerland, 2005.
[10] P. Golonka, "JCOP framework advanced course, access control component," Oct. 2006 [Online]. Available: http://itcobe.web.cern.ch/itcobe/Services/Pvss/Training/AdvancedCourse/Slides/advancedCourse-AccessControl.pdf
[11] KDE e.V., Darmstadt, Germany, "KIOSK," [Online]. Available: http://developer.kde.org/documentation/tutorials/kiosk/index.html
[12] CERN, Geneva, Switzerland, "The ATLAS TDAQ control and configuration," [Online]. Available: https://twiki.cern.ch/twiki/bin/view/Atlas/DaqHltControlAndConfiguration
[13] The OpenLDAP Foundation, Minden, NV, "OpenLDAP," [Online]. Available: http://www.openldap.org