Access Control Implementation in the ATLAS TDAQ

Marius Leahu, Giovanna Lehmann Miotto, Marc Dobson
from *ATLAS TDAQ Controls and Configuration*, and *ATLAS TDAQ SysAdmin*

Role Based Access Control Model

The Role Based Access Control(RBAC) model used in the ATLAS experiment[1] 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 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:

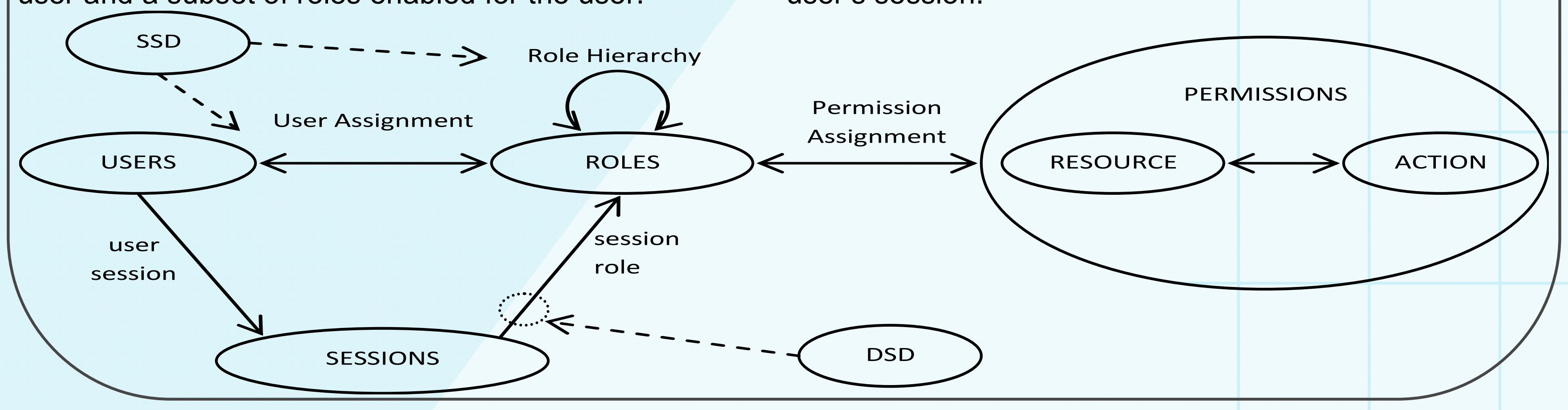
- USERS: human beings or automated agents;
- ROLES: job functions or job titles which define an authority level;
- RESOURCE: object which supports a set of possible ACTIONs;
- PERMISSIONS: approvals to perform an action on a given resource.

The sessions (SESSIONS) are mappings between a user and a subset of roles enabled for the user.

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 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. The DSD relation puts restriction on the roles that can be enabled within a user's session.



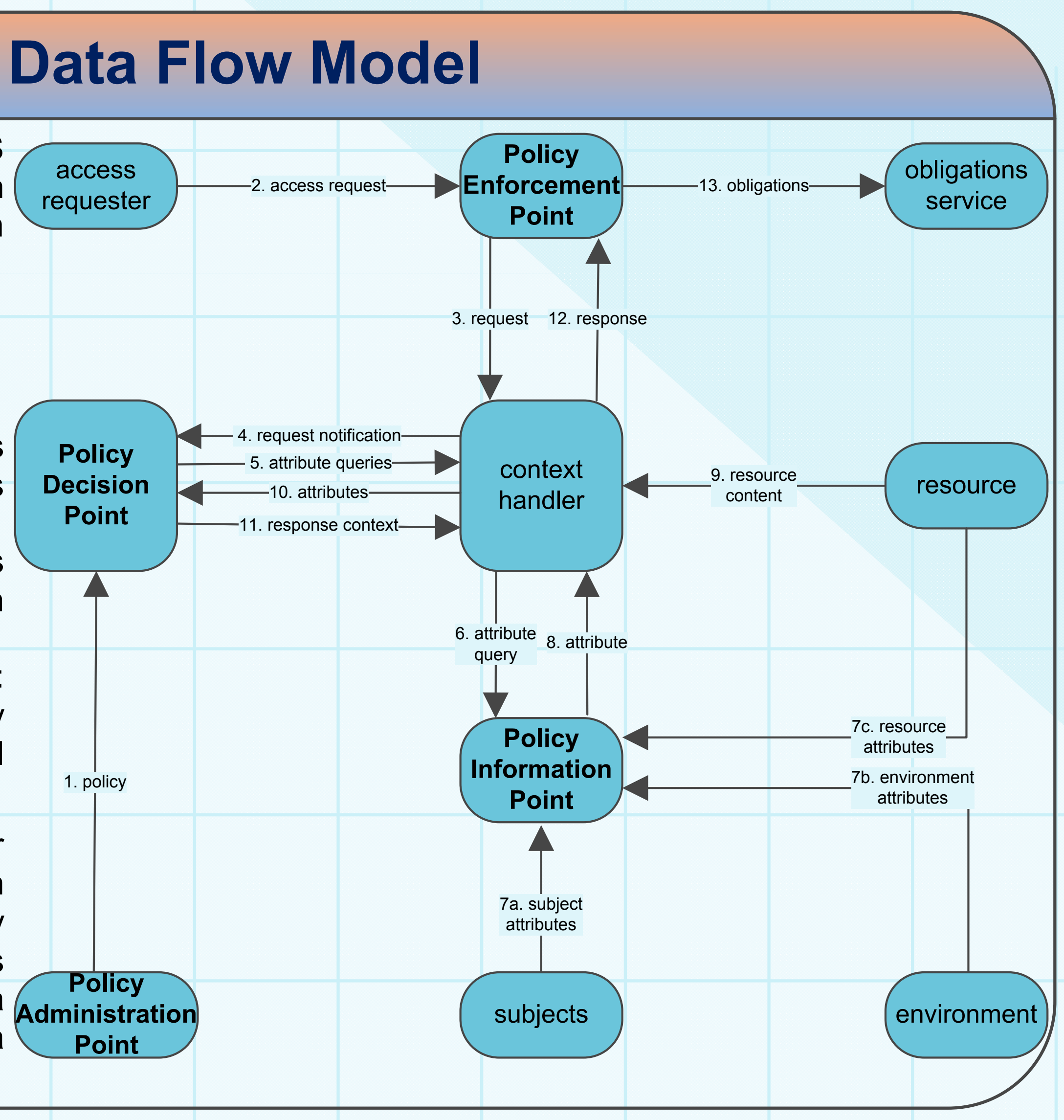
Data Flow Model

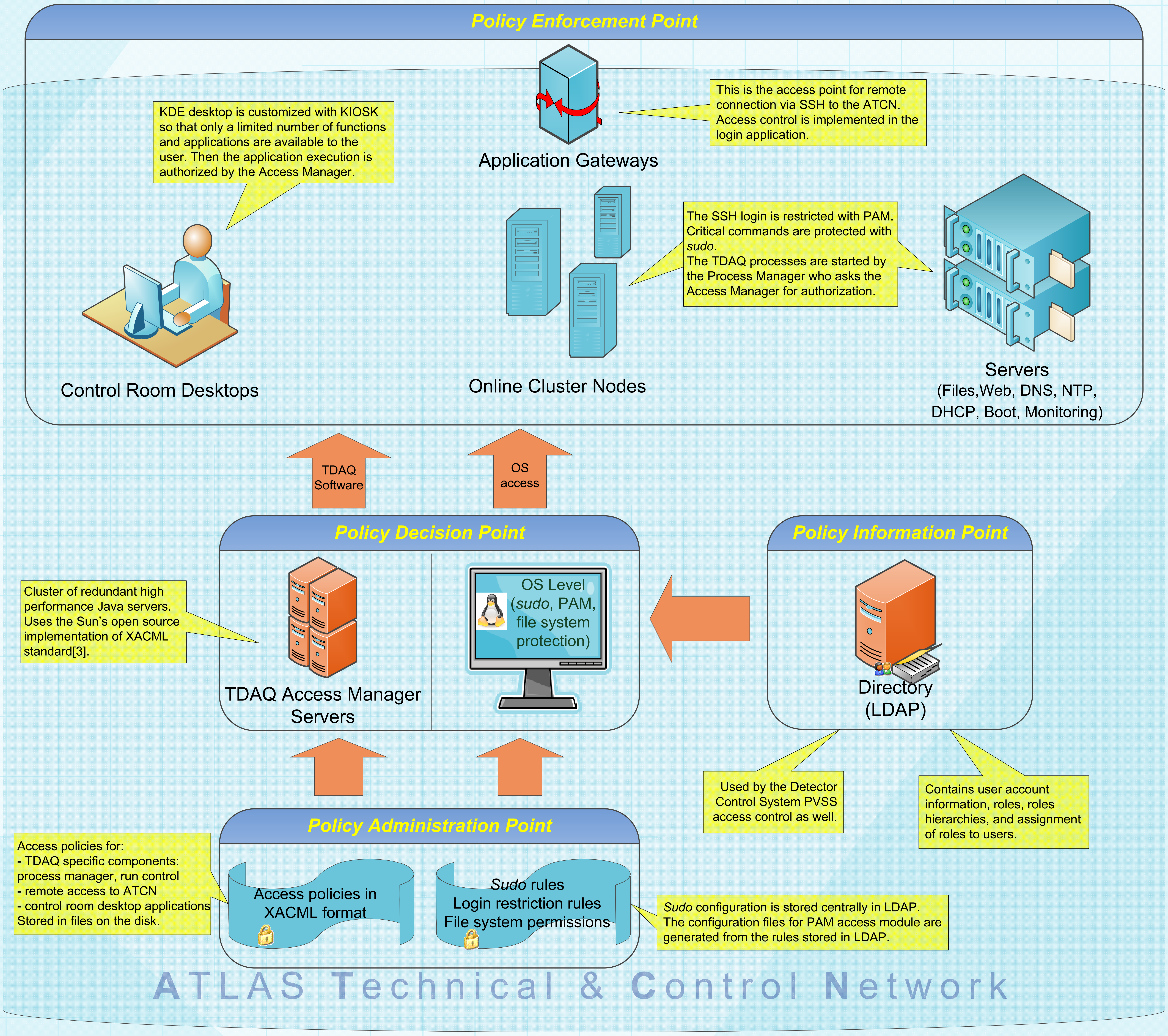
The data flow in the ATLAS TDAQ access control implementation is based on the data flow model from the OASIS XACML standard [2].

The main system entities are:

- **Policy Administration Point**: creates the policies.
- **Policy Information Point**: acts as a source of attribute values required in the policy evaluation.
- **Policy Decision Point**: evaluates applicable policy and renders an authorization decision.
- **Policy Enforcement Point**: performs the access control by making decision requests and enforcing authorization decision.

The access requester may be for example a user who wants to log in through the application gateway inside ATCN, or the TDAQ Process Manager application that stops a critical application running on a cluster node.





This is the access point for remote connection via SSH to the ATCN. Access control is implemented in the login application.

KDE desktop is customized with KIOSK so that only a limited number of functions and applications are available to the user. Then the application execution is authorized by the Access Manager.

The SSH login is restricted with PAM. Critical commands are protected with *sudo*.
The TDAQ processes are started by the Process Manager who asks the Access Manager for authorization.

Cluster of redundant high performance Java servers. Uses the Sun's open source implementation of XACML standard[3].

Used by the Detector Control System PVSS access control as well.

Contains user account information, roles, roles hierarchies, and assignment of roles to users.

Access policies for:
- TDAQ specific components: process manager, run control
- remote access to ATCN
- control room desktop applications
Stored in files on the disk.

Sudo configuration is stored centrally in LDAP. The configuration files for PAM access module are generated from the rules stored in LDAP.

References

[1] M.C. Leahu, M. Dobson, G. Avolio, "Access Control Design and Implementations in the ATLAS Experiment", *IEEE Trans. Nucl. Sci.*, vol 55, pp. 386-391, Feb. 2008

[2] OASIS XACML Standard. Available: http://www.oasis-open.org/specs/index.php#xacmlv2.0

[3] Sun's XACML implementation. Available: http://sunxacml.sourceforge.net

Contact

Marius Leahu	-CERN -Politehnica University of Bucharest	Marius.Leahu@cern.ch mleahu@alpha.imag.pub.ro
Giovanna Lehmann Miotto	-CERN	Giovanna.Lehmann@cern.ch
Marc Dobson	-CERN	Marc.Dobson@cern.ch