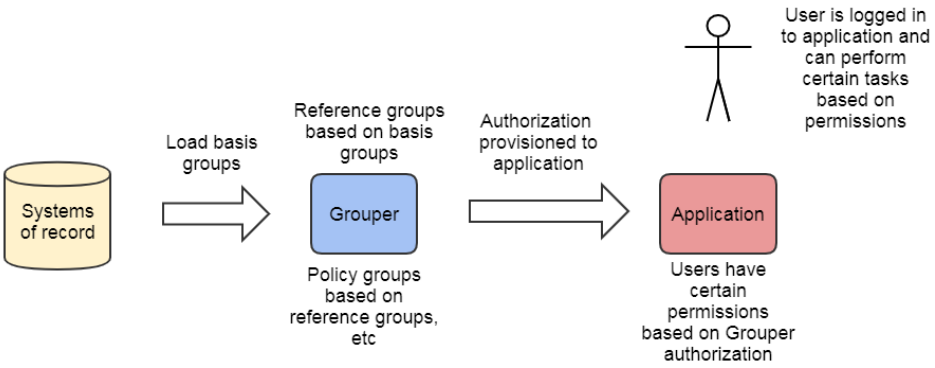
Access Control Models

Overview

The general Grouper approach to access governance is to create and maintain access policy groups which are built from institutional meaningful cohorts (reference groups), ad hoc exception groups, and indirect basis groups. Membership in the access policy group represents a pre-computed access policy decision. The policy decision is typically communicated to the target services via LDAP or SAML, but can be done in a variety of ways. To help implement these access control models, review the documentation on the Grouper Template Wizard.



The target service uses the policy decision from Grouper to determine what level of access the user has. The policy decision could be mapped to a hard-coded application role or a fine-grained permission set managed locally in the service. For example, the user might be dynamically assigned to a group in Confluence due to a SAML entitlement provisioned via Grouper. The user might then have access to projects due to configured access in the Confluence admin screens. This is explained in the "Policy groups and dynamic application permissions" model below.

Mechanisms to communicate and enforce policy decisions can vary considerably depending on the security needs and capabilities of the target service. However, the overall approach to access governance with Grouper remains consistent. The rest of this section uses terminology and models from NIST SP 800-162 and XACML to demonstrate a variety of models leveraging the overall approach. Note, you could use multiple models at once, or could engineer hybrid approaches.

Access Control Models

Most of the **Access Control Models (ACM)** lend themselves to distributed access control, meaning the authority to manage an access control policy or exceptions to policy can be delegated to authorized people. The most common ACMs are listed here, with a diagram and more details below on each one.

Access Control Model	Type (s)	Description	When to use	Notes
Policy groups and static application permissions	Policy groups	Roles are managed in Grouper and the permissions of those roles are hard-coded in the application or are opaque. This is a very common access control model.	If the application has hard-coded permissions based on application roles, and the roles can be provisioned from Grouper	See the U. of Chicago VPN access diagram /example and the examples on the Example Access Policies page.
Policy groups and dynamic application permissions	Policy groups	Very similar from a Grouper perspective to "Policy groups and static application permissions". Authorization configuration is in two places, who has which role (in Grouper), and what each role/user can do (in app)	If the application can configure permissions based on users/roles, and roles can be provisioned from Grouper	Two users in the same role might have different permissions if there are individual permission assignments in addition to role permissions assignments. See examples for Box and Duo.
Policy group for coarse-grained access	Policy group Coarse grained authn	Identify the population of who should be able to log in to the application, make a policy group, and lock out users not in that group	Use this for local or SaaS applications. This drastically improves your security posture. Use this if you cannot integrate roles with the application, or even if you can!	This can be setup as a reverse proxy to protect the application from any unauthenticated, unauthorized access. See the U. Penn PeopleSoft example.
Grouper for access reporting	Access reporting	Access is configured in the application. Assignments in the application are loaded into Grouper to help with reporting and deprovisioning	Use this if the roles/permissions in the application cannot be externalized to Grouper. Its very valuable for Grouper to track the assignments.	You can use Grouper attestation, Grouper deprovisioning, etc. on read-only access assignments

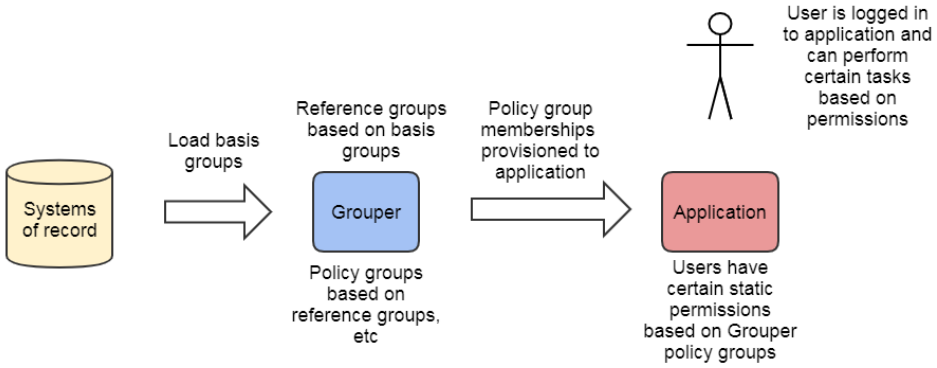
Grouper managed permissions	Policy groups Externalized permissions	Policy groups are used as roles in Grouper and permissions are assigned in Grouper to the roles /users. Permissions are provisioned to the application	For custom application where you want to offload the permissions to Grouper as an RBAC engine. Its possible though less common for packages as well.	The difficulty can be if the app can support externalize permissions, if the grouper RBAC permission model fits the application, and if the generic Grouper permission screens are usable for the application
eduPerson Affiliation for authorization	Reference groups	Use eduPersonAffiliation or equivalent to secure access e.g. based on if the user is a student or employee	Use this if the application (SaaS?) only supports security by eduPersonAffiliation	Less mature, less flexibility, not recommended Use only as a last resort You could work around this it might not be ideal Lacking delegation

The following sections provide a diagram and more detail on each of the ACMs listed in the table above.

ACM Policy groups and static application permissions

Like many ACMs that use policy groups, access policy administration and the policy decision point (pre-computed membership assignments) are done in Grouper and can be communicated to the target service in a variety of ways. Access policy groups in Grouper enable direct mapping from natural language policy to digital policy and back, and policy is kept up to date automatically as subject attributes change. Grouper policy groups support audit, compliance, and attestation. This model can be broadly used with a variety of services.

1. Access policy groups configured in Grouper based on institutional meaningful cohorts (i.e. reference groups, ad hoc groups, etc)
2. Authorization provisioned to application via LDAP, SAML, web services, direct provisioning, etc
3. Application uses those role(s) with hard-coded or static permissions to allow the user to perform actions

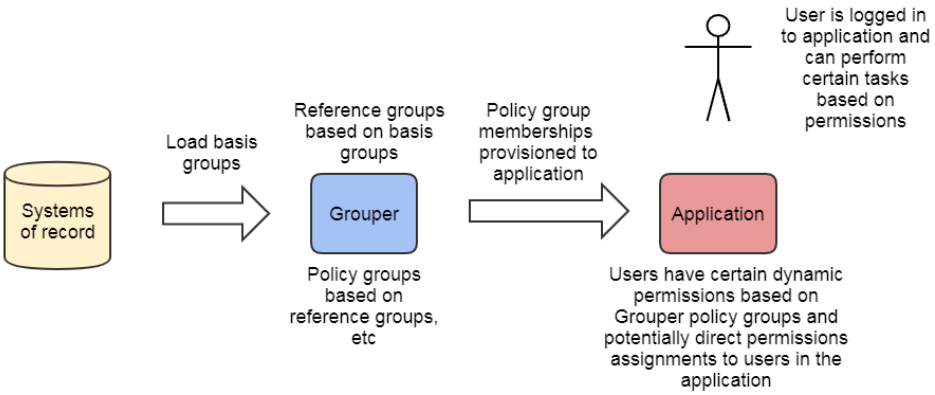


ACM Policy groups and dynamic application permissions

In applications with sophisticated RBAC capabilities, fine-grained permission sets are typically configured via an administrative interface within the application itself. These permission sets are then associated with a role that can be mapped to a set of users. In this model, the user to role mapping is done in Grouper by pairing a normal access policy group with the role defined at the target service. The policy of which subjects are mapped to application roles is similar to other ACMs with policy groups.

1. Access policy groups configured in Grouper based on institutional meaningful cohorts (i.e. reference groups, ad hoc groups, etc) provides User -> Role mapping
2. Authorization provisioned to application e.g. with LDAP, SAML, WS, direct provisioning, etc
3. Fine-grained permission sets are managed at the target service (Role -> Permissions and User Permissions)

If permissions are also assigned to individual users using the the application's permission management interface (not necessary for this ACM), then Grouper's view of what a user has is not necessarily the same as the application's. Two users who have the same role in Grouper could have different permissions in the application. Grouper knows at a high level what access the user has, but you would need to consult the application for the full picture.



ACM Policy group for front door access

Often for pragmatic reasons one wants to limit access to a target service by preventing authentication from completing or by preventing network traffic from reaching the application. This "front door" access control provides a logical outer perimeter in addition to the applications normal authorization mechanisms. This could be used in situations where the target application does not have sufficient access controls either due to technical or administrative reasons, or to provide another security layer.

In cases where it is preferable to completely limit unauthorized network traffic from reaching the application, a reverse http proxy can be used block unauthorized and possibly nefarious network traffic. Combing "front door" access with other ACM models enables defense-in-depth. Several cases where this is necessary or desirable are; 1) when the target service has insufficient access controls to limit access based on the desired policy, 2) when the target service lacks a good unauthorized user experience, and 3) when the application runs on software that might have 0-day exploits and the security patches are difficult to keep up with.

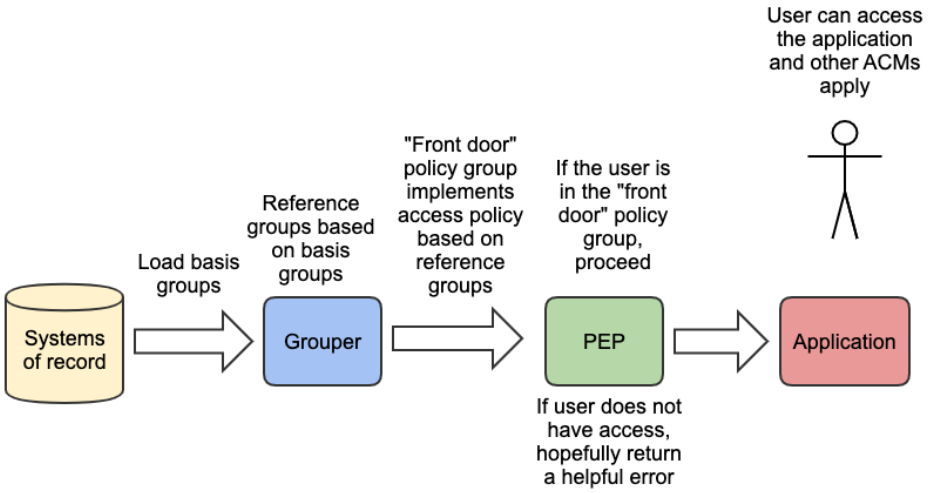
The "front door" policy group membership could be wider than what is actually authorized to use the target application, if the application additionally provides its own authorization (e.g. identify the admin users). So at a minimum the policy group could contain an affiliation (e.g. "employee" reference group), even if only certain employees have access to the application. The most precise "front door" policy group would contain only the subjects that have access to the application, either via Grouper policy groups or application specific configuration.

There are several options for the "front door" ACM **policy enforcement point (PEP).**

PEP	Description	When to user	Notes
SAML Identity Provider	The Shibboleth IdP can block access or send a blank assertion if the authenticated user is not in a certain Grouper policy group when authenticating to a certain Service Provider (SP)	SaaS services Any SP integrated with your IdP	There might or might not be a friendly error page Some IdP operators might not to mix authentication with authorization on principal Some SaaS apps might allow SAML authentication in addition to local "back door" authentication, so it might not provide the sufficient intended protection in those cases (e.g. box) You could mine IdP logs to see what population has recently used an application to make sure you are not making the front door policy group too restrictive
Load balancer (e.g. F5)	Application load balancers that reverse proxy http traffic might be able to limit traffic by a subject attribute or group membership. For example, the F5 load balancer has been successfully integrated with SAML, and can restrict traffic to an application by entitlement from Grouper. This could be a security reverse proxy component as well, or a CASB (Cloud Access Security Broker).	Any application behind a load balancer that authenticates the user.	The application needs to be reverse proxiable The UI (SAML) traffic needs to be separate from any WS or public traffic Blocking all network traffic by a reverse proxy improves the threat model since attackers must be authenticated and authorized Instead of a load balancer, the component could be an application firewall
Web server (e.g. Apache)	Web servers can reverse proxy traffic or just serve pages and can block all network traffic unless someone is authenticated and authorized	Any application reverse proxied behind an apache Any apache application	The application needs to be reverse proxiable if using apache as reverse proxy The UI (SAML) traffic needs to be separate from WS or public traffic Blocking all network traffic by a reverse proxy improves the threat model since attackers must be authenticated and authorized

Service provider (SAML SP)	The SAML SP can block authentications if not in a certain Grouper group	Any application that uses a SAML SP	It's possible that nefarious traffic could attack the application since the SP is generally along side the application and not in front
"Can login" role	The application might have a group that indicates a user has access (e.g. jira-users)	If the application has a "can login" group	The provisioned list of accounts, or a role of "can login" or "user" is similar to a coarse-grained security control This does not help secure network traffic like a reverse proxy

1. A "front door" access policy group is configured in Grouper that indicates who can use an application
2. Authorization provisioned (e.g. LDAP, SAML, etc) to the enforcement point (e.g. IdP, F5, Apache)
3. If a user is not authorized they should be directed to the "not allowed for this app" page
4. The application should do its own security using another ACM (e.g. get roles from other Grouper policy groups)



ACM Grouper for access reporting

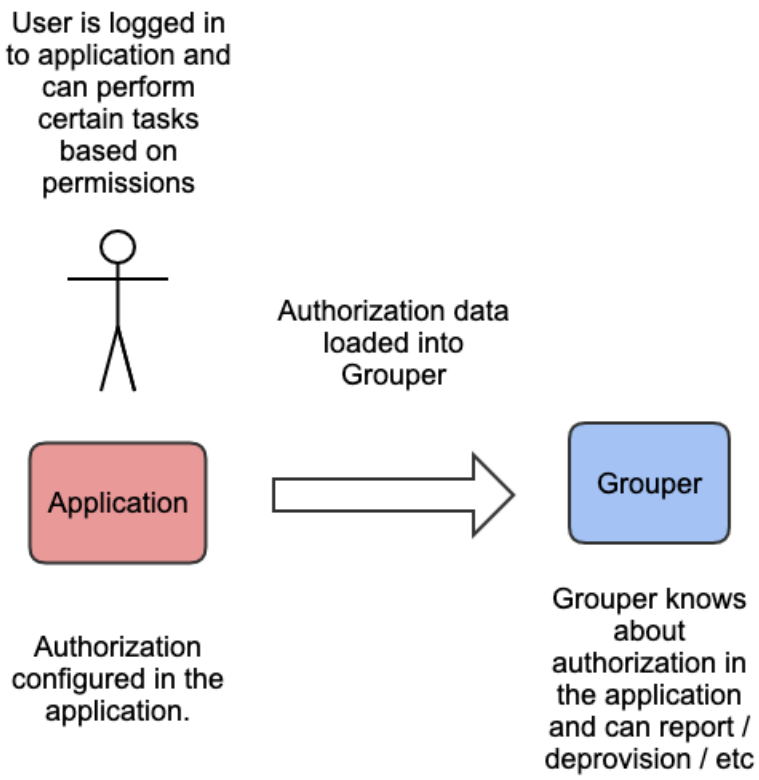
It is ideal if Grouper can be system of record for authorization for an application, but in many cases it is not possible. This could be because the application cannot use external roles, or because the application is preferred to manage those things internally (e.g. the UI in the application is needed for use). In this case hopefully you can feed authorizations from the application, or just who has any authorization, into Grouper. Now when you go to Grouper and see what someone has access to, you can see that application in the list.

Loading application authorization into Grouper needs to be a regularly schedule task so the data does not diverge. It could also be loaded near real-time. Several options exist for import authorization data:

1. Grouper loader via SQL or LDAP (preferred)
    a. You could ETL the data to a SQL or LDAP first
2. Web services
3. Custom job
4. Manual process (e.g. export the authorizations to CSV and import into Grouper. Do this periodically (e.g. monthly)

Once the authorizations are in Grouper you can use Grouper reporting, rules, attestation, deprovisioning, auditing, etc. This data in Grouper is readonly since it is sourced and managed in the application. In this case, deprovisioning or attestation must involve a user using the application to remove the unnecessary authorizations.

1. Authorizations are managed in the application
2. Authorizations are loaded into Grouper
3. Deprovision, attest, and report in Grouper by making security changes in the app

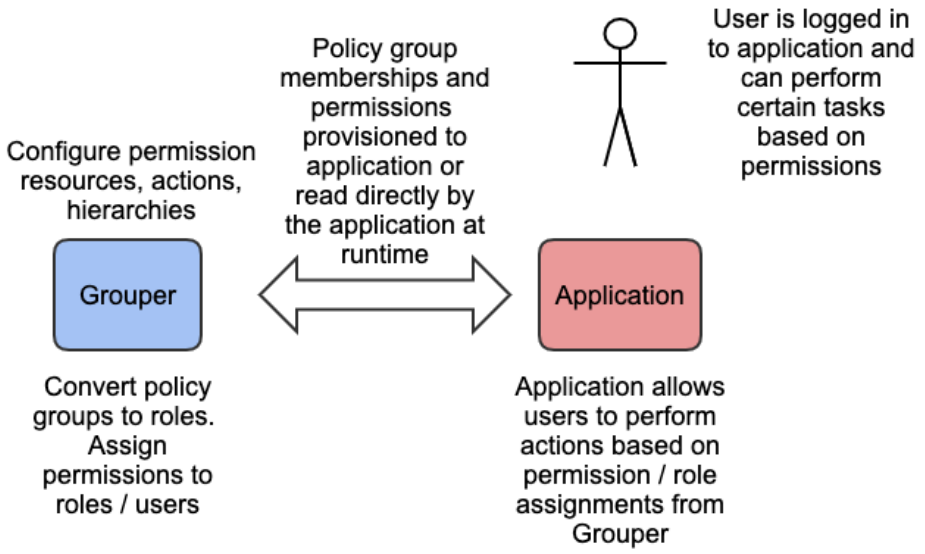


ACM Grouper managed permissions

While mostly out of scope for this guide, Grouper does have an advanced RBAC capability that is suitable for externalizing application permission management. In this model, the target system relies on Grouper for application permission management, including permission definition, role and resource hierarchies, and role to permission mapping.

Grouper role and permission management provides more details on this advanced use of Grouper.

1. Create policy groups similar to other ACMs
2. Convert those groups into roles
3. Configure permission resources and assign to roles / users (in Grouper)
4. Provision roles/permissions to application. Note, something more complex than LDAP/SAML is generally used, e.g. WS or SQL

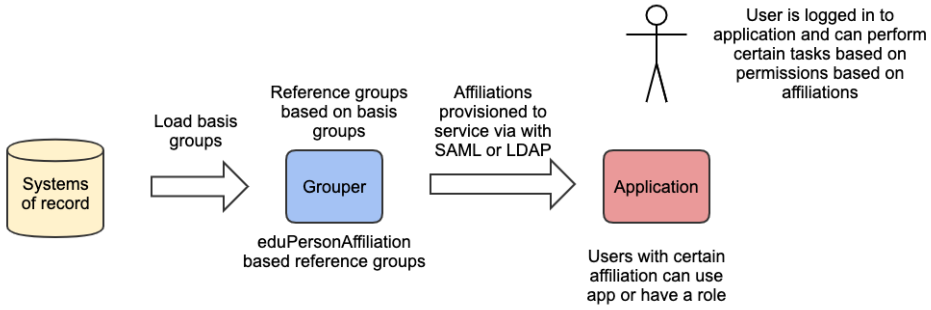


ACM eduPersonAffiliation for authorization

In this model, Grouper is used to master subject attributes that represent some type of affiliation or status at the institution. Generally these are reference groups for eduPersonAffiliation: member, student, employee, etc. Actual access policy administration is completely local to the target service. This model was more common years ago before sophisticated authorization was common. It is not recommended to do this since you are back to reference groups without the flexibility of policy groups.

This model is useful for cases when there is an informal relationship between the institution and the service provider, and a locally defined notion of the subject attribute like eduPersonAffiliation is sufficient for access control. However, the model breaks down quickly if a more exact notion of the subject attribute is required or if it needs to be different across services. It is important to remember that cohorts (affiliations, status, class year, etc) are not access policy. Do not be tempted to create service specific versions of reference groups. Favor creating access policy per service instead. Any application using this model should be comfortable with eduPersonAffiliation standard's notion of "broad-category affiliation assertions".

1. Subject attributes like eduPersonAffiliation are mastered in Grouper
2. Affiliations provisioned to the application typically via SAML Authentication Response
3. The service allows access or uses the affiliation to map to an application role / permission set



Previous: Folder and Group Design

Next: Provisioning Models