When Roles Aren't Enough

Modern Permissions in Drupal with the Access Policy API

Michael Harris @miwayha

Access Policy API: A Brief History

Limitations of Roles and Permissions

Limitations of Roles and Permissions

1. Problems with Too Few Roles
2. Problems with Too Many Roles
3. Problems with Static Permission and Custom Access Logic

Static Permission and Custom Access Logic

Your business requirements might might mean access is based on context:

- time
- session attributes
- user attributes
- entity values
- configuration

Your custom access logic is hard to find in your code base:

- route _custom_acces
- form alters
- node grants
- hook_entity_access()
- hook_entity_field_access()
- render array #access
- menu plugin isEnabled()

Meet the Access Policy API

Meet the Access Policy API

\Drupal\Core\Session\AccessPolicyInterface::calculatePermissions()

- Adds permission to a user
- Place to add Cache tags, max-age

\Drupal\Core\Session\AccessPolicyInterface::calculatePermissions()

- *Remove* permissions from a user
- Place to add Cache tags, max-age

\Drupal\Core\Session\AccessPolicyInterface::getPersistentCacheContexts()

- Place to add cache contexts

Demos

Should I Use the Access Policy API

You Already Are

User::hasPermission (Drupal < 10.3)

```
public function hasPermission($permission) {
    ...
    return $this->getRoleStorage()
        ->isPermissionInRoles($permission, $this->getRoles());
}
```

User::hasPermission (Drupal > 10.3)

```
public function hasPermission($permission) {
  ...
  return \Drupal::service('permission_checker')
        ->hasPermission($permission, $this);
}
```

core.services.yml (Drupal > 10.3)

```
permission_checker:
  class: Drupal\Core\Session\PermissionChecker
  arguments: ['@access_policy_processor']
```

core.services.yml (Drupal > 10.3)

```
access_policy.user_roles:
  class: Drupal\Core\Session\UserRolesAccessPolicy
  arguments: ['@entity_type.manager']
  tags:
   - { name: access_policy }
```

But Maybe You Should Write Your Own

Use Cases

- You have a lot of authenticated users with different access needs
- Your access control needs vary based on time, location, configuration, entity values, user attributes, etc.
- You can model your access needs with permissions, but things start to get fuzzy when you switch from permissions to roles
- You're comfortable with cache tags, contexts, and max-age

Final Thoughts

Modules that use the Accecss Policy API

- Group
- External Roles
- Inline Permissions

Resources

- Introducing the new Access Policy API in Drupal Core - DrupalCon Barcelona 2024
- New access policy API Change Record - Drupal.org
- Super user access policy can be turned off Change Record - Drupal.org
- Access Policy API Documentation - Drupal.org
- Episode #472 - Talking Drupal
- Drupal Access Policy API demystified - Luca Lusso (SparkFabrik)
- api.drupal.org:
  - AccessPolicyInterface
  - AccessPolicyBase