OACC

High-performance, Transactional, Java Security Framework

*www.**oacc**framework.org*

Adinath Raveendra Raj

PHXJUG presentation
October 8, 2014

What is OACC?

OACC is an acronym for **O**bject **A**ccess **C**ontrol servi**C**e

OACC defines a **security model** and provides an **API** that implements the model

OACC = security model + API

Brief History

- First encountered and solved the problem of object security in 2002
- Work on the first version of OACC started late 2009
- OACC 1.0 was completed in the Spring 2010
- OACC 1.0 went into production in 2010
- Received a number of requests to make OACC open-source, last two were in the Summer of 2014
- Started work on enhancements to OACC in Summer 2014
  - Comprehensive test suite
  - Refactor implementation
  - Add support for Oracle and at least one other major DBMS
  - Examine the API for ease-of-use

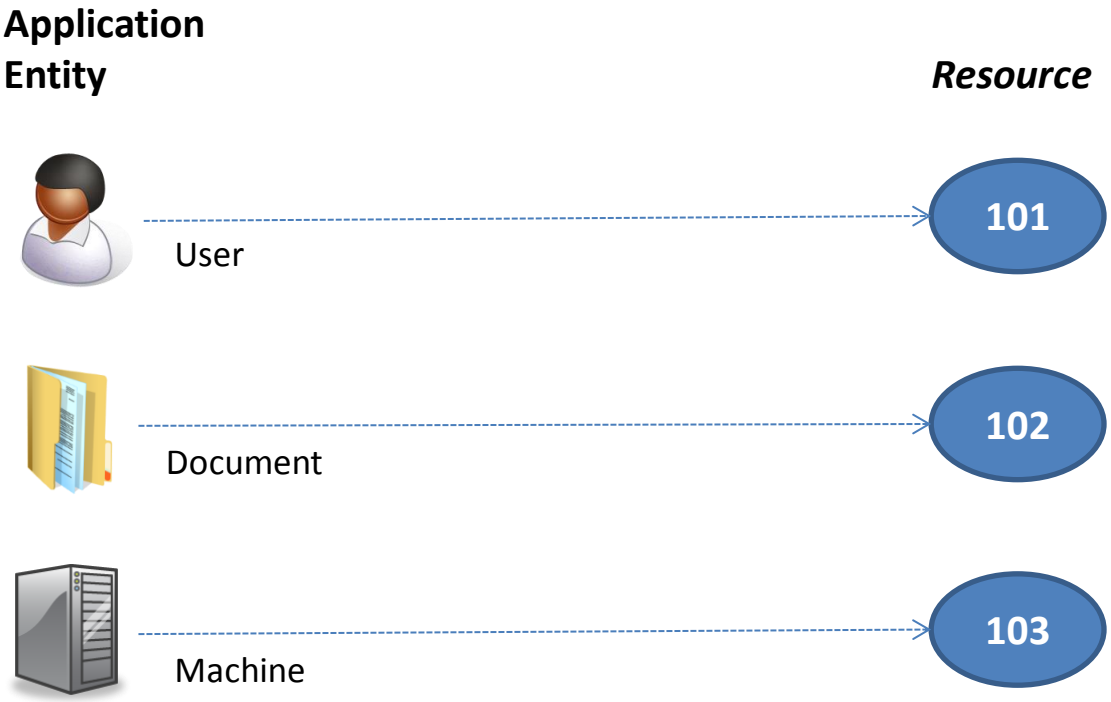
Overview

- **define**
  - Define resource classes
  - Define permissions for each resource class
- **identify**
  - Create resources
  - Create resource domains
- **authenticate**
  - Password
  - Impersonate
  - External provider
- **authorize**
  - Permissions based model
    - Built-in system permissions
    - Application specific permissions
  - Fine-grained – resource scope
  - Coarse grained – domain scope
- **query**
  - Assert permissions
  - Find resources based on permissions

- **define**
  - Define resource classes
  - Define permissions for each resource class
- **identify**
  - Create resources
  - Create resource domains
- **authenticate**
  - Password
  - Impersonate
  - External provider
- **authorize**
  - Permissions based model
    - Built-in system permissions
    - Application specific permissions
  - Fine-grained – resource scope
  - Coarse grained – domain scope
- **query**
  - Assert permissions
  - Find resources based on permissions

What is a ***Resource***?

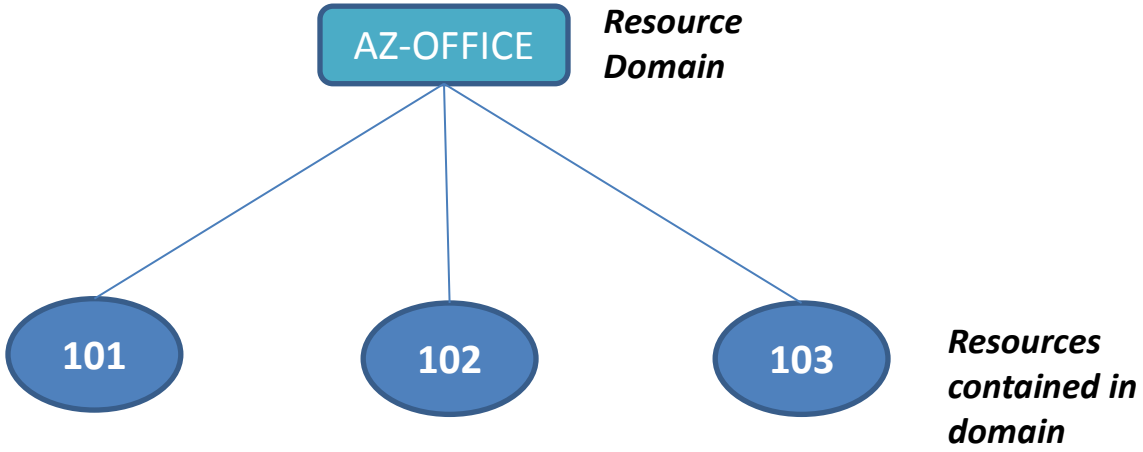
A **resource** represents an **application object**. An application object must **persist** the **resource identifier** for its associated resource to integrate with OACC.



What is a ***Domain***?

A **domain** is,

- identified by its string **name** (e.g. “AZ-OFFICE”)
- a **container** for resources (every resource is in **exactly one** domain)
- serves as a mechanism to **address** a large number of **logically related resources**



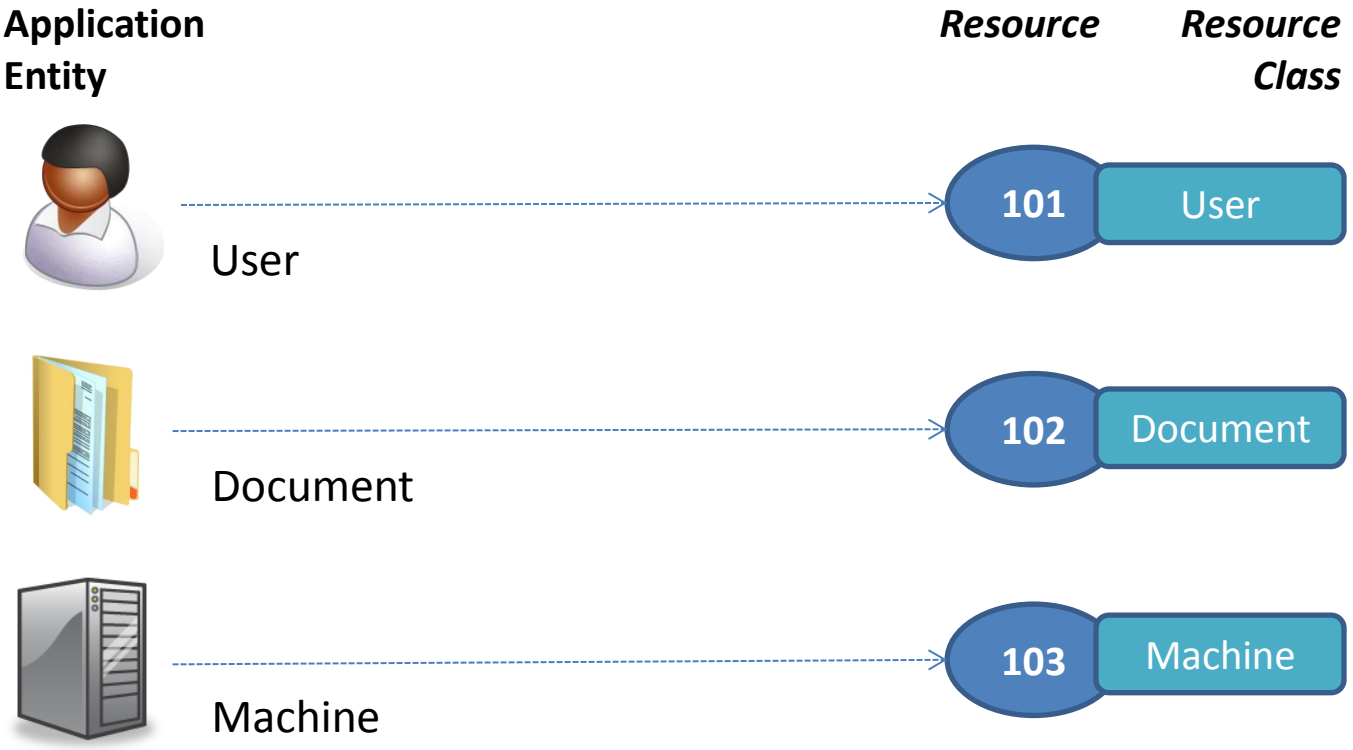
Example: In a multi-tenant system, each tenant could have its own domain

- **define**
  - Define resource classes
  - Define permissions for each resource class
- **identify**
  - Create resources
  - Create resource domains
- **authenticate**
  - Password
  - Impersonate
  - External provider
- **authorize**
  - Permissions based model
    - Built-in system permissions
    - Application specific permissions
  - Fine-grained – resource scope
  - Coarse grained – domain scope
- **query**
  - Assert permissions
  - Find resources based on permissions

What is a ***Resource Class***?

A **resource class** is,

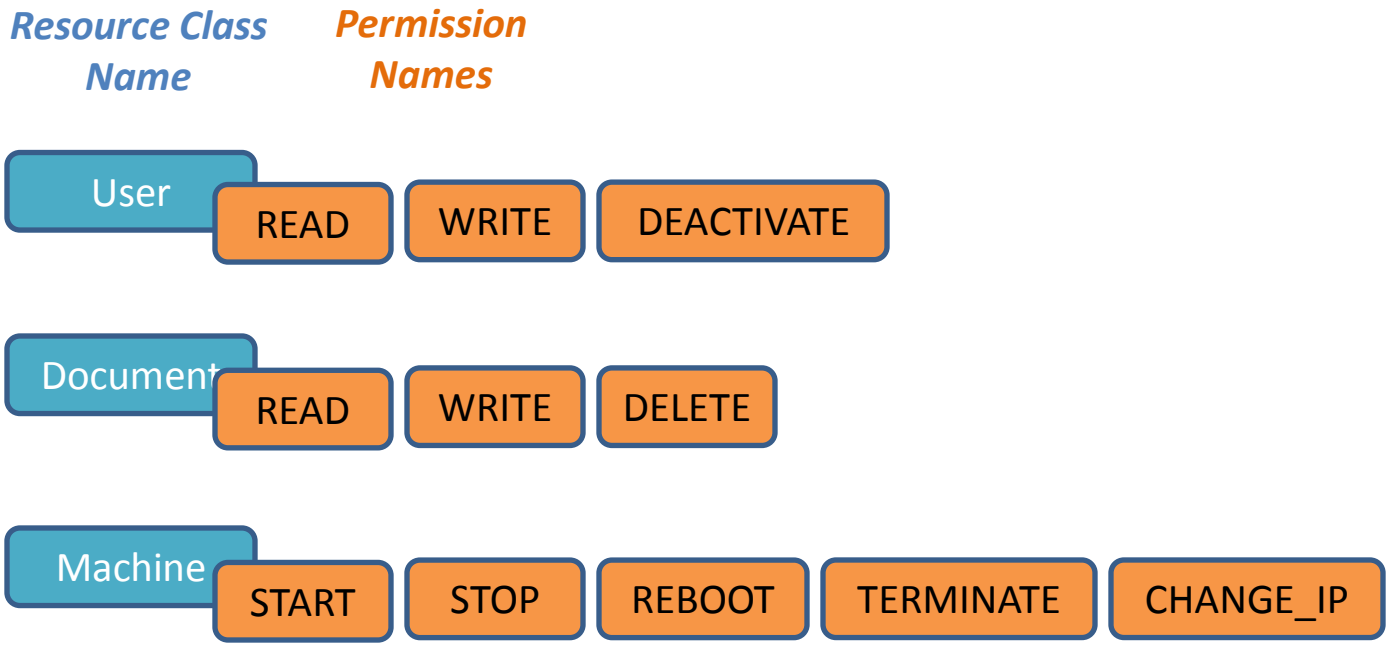
- identified by its string **name** (e.g. “DOCUMENT”)
- defines **a set of permissions names** applicable to resources that belong to the class
- is a **required argument** in the API call **to create a resource**



What is a ***Permission***?

A **permission** is,

- identified by its string **name** (e.g. “READ”)
- **must be paired with a resource class** to make sense



- **define**
  - Define resource classes
  - Define permissions for each resource class
- **identify**
  - Create resources
  - Create resource domains
- **authenticate**
  - Password
  - Impersonate
  - External provider
- **authorize**
  - Permissions based model
    - Built-in system permissions
    - Application specific permissions
  - Fine-grained – resource scope
  - Coarse grained – domain scope
- **query**
  - Assert permissions
  - Find resources based on permissions

Authentication

- Before any OACC API calls can be the API **caller** must **authenticate** with **API session**.

- The caller authenticates by providing its resource identifier and password to the API session. If successful the resource provided becomes the **authenticated resource**.

- Authentication alone does not grant any privilege. An API call succeeds only if the **authenticated resource** has the **authorization** to complete the requested operation.

- **define**
  - Define resource classes
  - Define permissions for each resource class
- **identify**
  - Create resources
  - Create resource domains
- **authenticate**
  - Password
  - Impersonate
  - External provider
- **authorize**
  - Permissions based model
    - Built-in system permissions
    - Application specific permissions
  - Fine-grained – resource scope
  - Coarse grained – domain scope
- **query**
  - Assert permissions
  - Find resources based on permissions

Authorization

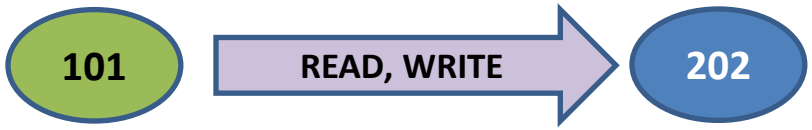
Authorization is essentially answering the question:

> Does the **accessor resource** have the **requested permissions** to the **accessed resource?**

Next we look at the different ways an accessor resource can have permissions to an accessed resource.

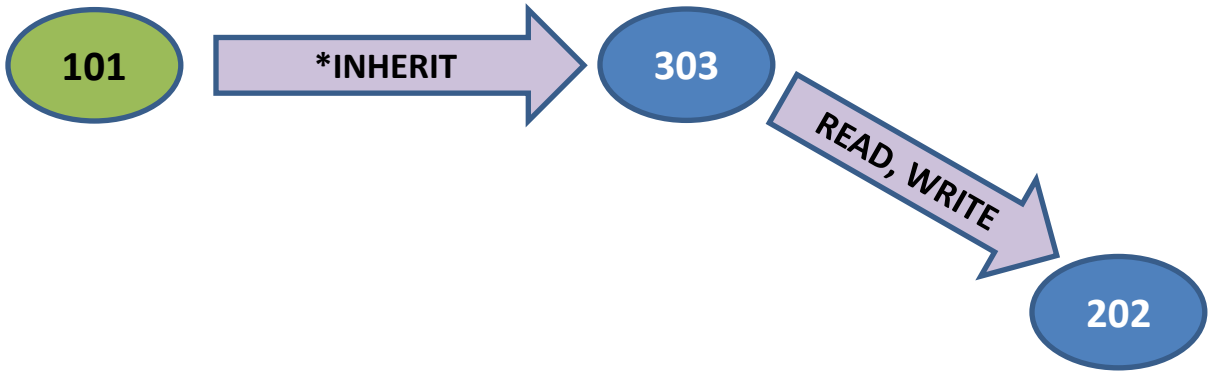
Direct Permissions (Resource)

- The **accessor** resource has **direct permissions** on the **accessed** resource.



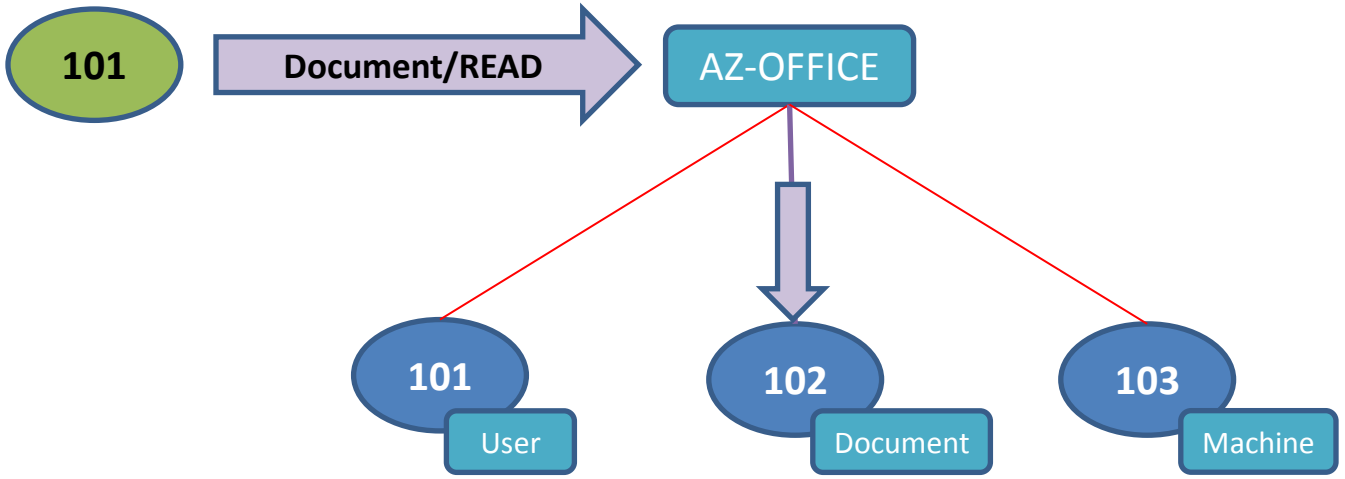
Inherited Permissions (Resource)

- The **accessor** resource has **no direct permissions** on the **accessed** resource
- The **accessor** resource has **inherit** permission on a **third resource** that has **direct permissions** to the **accessed** resource.



Global Permissions (Resource)

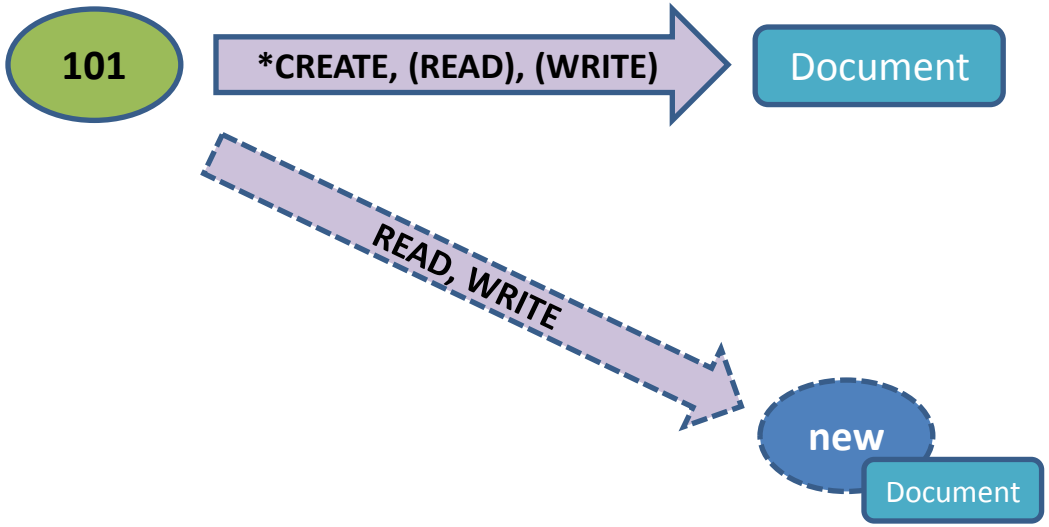
- The **accessor** resource has permissions on **all** resources of a **specified resource class** in a **specified domain**.



Create Permissions (Resource)

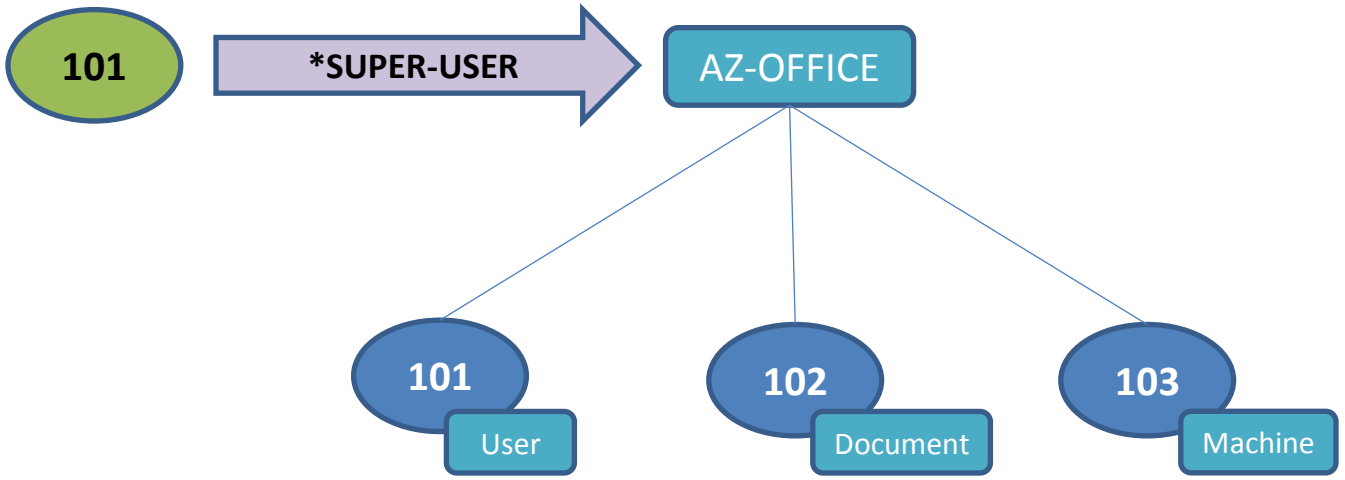
- The **accessor** resource has permissions to **create** resources of a **specified resource class** in a **specified domain**.

- Optionally **post create permissions** can be specified. Post create permissions **are permissions the accessor gets on new resources** it creates (READ, WRITE in the example below).



Direct Permissions (Domain)

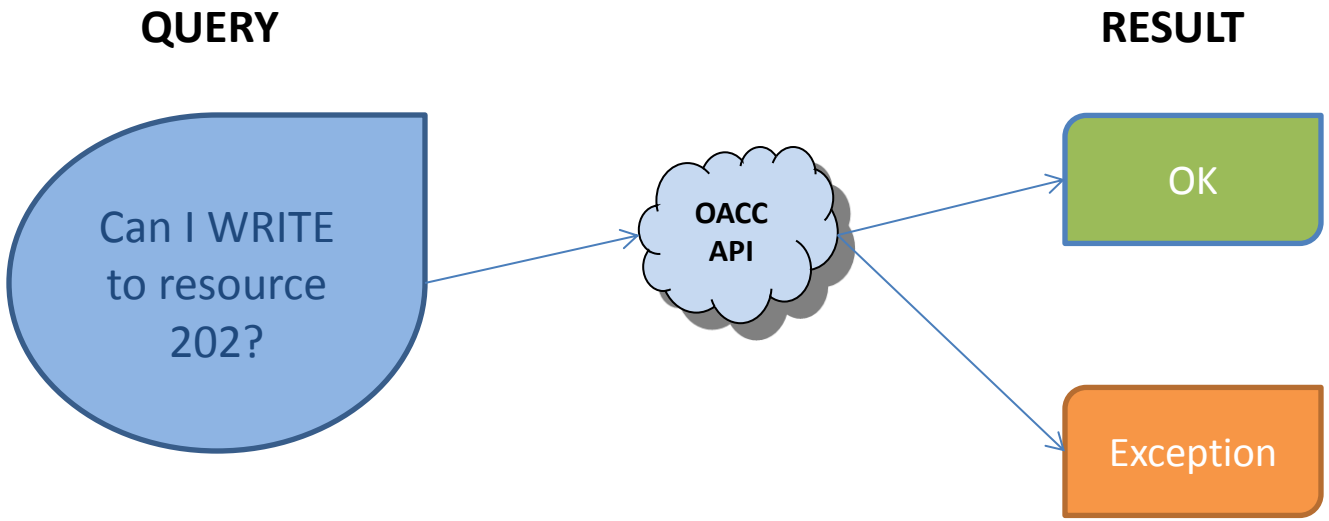
- The **accessor** resource has **direct permissions** on the **accessed** domain.



- **define**
  - Define resource classes
  - Define permissions for each resource class
- **identify**
  - Create resources
  - Create resource domains
- **authenticate**
  - Password
  - Impersonate
  - External provider
- **authorize**
  - Permissions based model
    - Built-in system permissions
    - Application specific permissions
  - Fine-grained – resource scope
  - Coarse grained – domain scope
- **query**
  - Assert permissions
  - Find resources based on permissions

Checking permissions

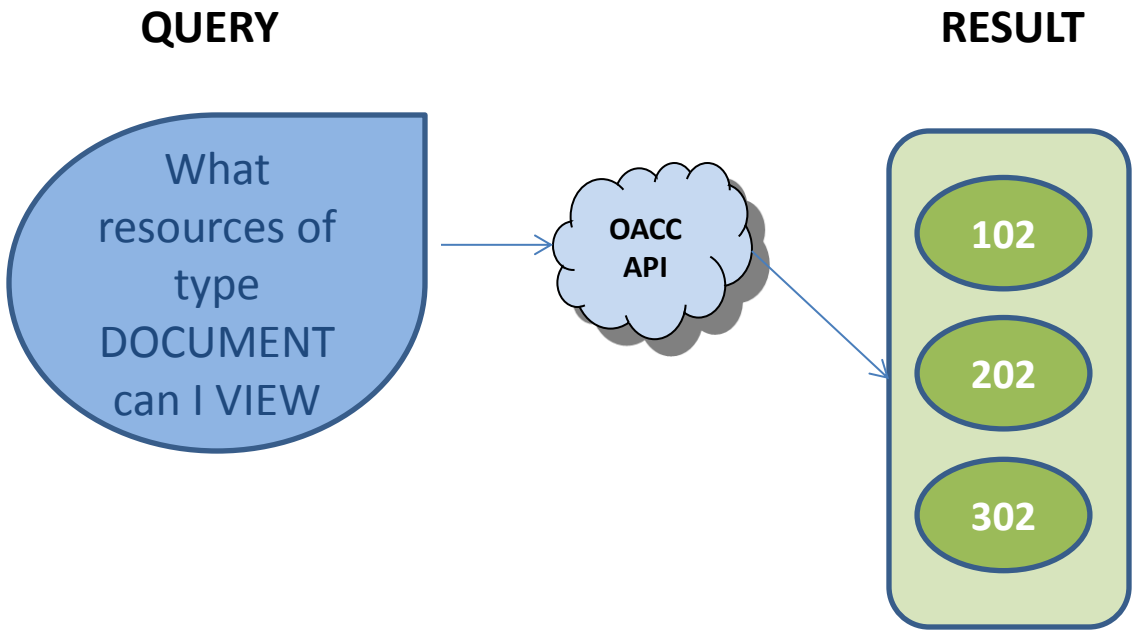
Check if the **accessor** has a specified **permission** to a **resource** (**accessed**)



void **assertResourcePermission**(Resource **accessedResource**,
ResourcePermission **resourcePermission**)

What is resource areavailable?

Check if the **accessor** has a specified **permission** to a **resource** (**accessed**)



Set<Resource> **getResourcesByResourcePermission**(String **resourceClassName**,
ResourcePermission **resourcePermission**)

Q & A

Project website: http://oaccframework.org

Sources: https://github.com/acciente/oacc