

# Lesson 2 Introduction to Permission Configuration

## 2.1 Requirements

Enigma Fire & Casualty Insurance would like to add some restrictions on which users can edit the Account Locations. Configure the Account Locations in PolicyCenter such that only users with the proper permission can edit the account locations, for example, producers and underwriters.

## 2.2 Configuration



### Activity

#### 1. Verify permissions for Account Location Information in the application before configuration.

The current behavior for account locations is that any users can edit account location information.

- a) Log in as different users. Verify that they all can edit account location information for the account Ray Newton.
  - aapplegate (underwriter)
  - aarmstrong (producer)
  - aauditor (auditor).

#### 2. In Studio, create the following system permission to edit account locations:

Code	Name	Description
editlocation_Ext	Edit account locations	Permission to edit locations on an account

- 3. Configure the AccountLocationPopup.pcf page so that only users with the appropriate system permission can edit the page.
- 4. Restart the server as required by additions to a typelist.



### Tip

#### Invalidate caches

If you still don't see the new typecode in Studio, then you might need to invalidate your caches and restart Studio.

File → Invalidate Caches / Restart → Invalidate and Restart



## Tip

### Generate dictionary

The security dictionary must be regenerated after a typelist is modified. It also helps to validate any entity and typelist changes. Command = `gwb genDataDictionary`

5. In PolicyCenter, login as su/gw.
6. Grant the new edit account location permission to the Producer and Underwriter roles.

## 2.3 Verification

1. Log in to Guidewire PolicyCenter as the following users and repeat the steps.
  - o Alice Applegate (an underwriter) with `aapplegate/gw`
  - o Archie Armstrong (a producer) with `aarmstrong/gw`
  - o Adam Auditor (a premium auditor) with `aauditor/gw`
  - a) Navigate to the Account Ray Newton.
  - b) Click on Locations in the side bar to display the Account File Locations list.
  - c) Click on one of the hyperlinks (Loc.#, Location Code and Location Name columns) associated with one of the locations.
2. Results
  - o User Alice Applegate and Archie Armstrong can edit the Account Location information in the popup.

Location Information		<a href="#">Return to Account File Locat</a>
Non-Specific Location	No	
Location Code	<input type="text" value="p001"/>	
Location Name	<input type="text" value="Location 0001"/>	
Country	<input type="text" value="United States"/>	
Address 1	<input type="text" value="0001 Bridgepointe Parkway"/>	
Address 2	<input type="text"/>	
Address 3	<input type="text"/>	
City	<input type="text" value="San Mateo"/>	<input type="button" value="📍"/>
County	<input type="text" value="San Mateo"/>	

- User Adam Auditor cannot edit the Account Location information in the popup.

Location Information
[Return to Account File Locations](#)

Non-Specific Location	No
Location Code	0001
Location Name	Location 0001
Address	0001 Bridgepointe Parkway San Mateo, TX 94404-0001



## 2.4 Solution



1. In Studio, modify `SystemPermissionType.tttx` to add a new typecode.

Code	Name	Description
editlocations_Ext	Edit account locations	Permission to edit locations on an account

SystemPermissionType.tttx					
category		Show All		SystemPermissionType.tti	1-2
Element	Code	Name	Priority	Name	Value
typecode	editlocations_Ext	Edit account locat...	-1	code	editlocations_Ext
typecode	internaltools	All internal tools	-1	name	Edit account locations
typecode	toolsInfoview	View Info tools pa...	-1	desc	Permission to edit locations on an

2. Configure AccountLocationPopup.pcf, so that only users with the appropriate system permission can edit the page.

Properties: Properties Variables Entry Points Code

Popup

Basic properties

canEdit	shouldEdit && perm.System.editlocations_Ext
canVisit	
id*	AccountLocationPopup

3. Restart server
4. In PolicyCenter, login as super user su/gw.
5. Go to Administration → Users & Security → Roles
6. Grant the new edit location permission to the Producer and Underwriter roles.

Edit → Add → Update

Role: Producer Up to Roles

Type User Producer Code Role

Internal Role Only \* ☒ Yes ☐ No

Description Permissions for producer

Permissions Add Remove

<input type="checkbox"/>	Permission*	Code	Description
<input type="checkbox"/>	Account Holder Info	viewaccountholderinfo	Permission to navigate to and view Account H
<input type="checkbox"/>	Advance cancellation	advancecancellation	Permission to advance a cancellation
<input type="checkbox"/>	Advance issuance	advanceissuance	Permission to advance an issuance
<input type="checkbox"/>	Copy job	jobcopy	Permission to copy a job
<input type="checkbox"/>	Copy policy data	copypolicydata	Permission to copy entities from one Policy to
<input type="checkbox"/>	Edit account locations	editlocations_Ext	Permission to edit locations on an account

## 2.5 References

### 2.5.1 Creating new system permissions

System permissions are defined in the `SystemPermissionType` typelist. Add typecodes to create new system permissions.



## Best Practice

Typecodes for `SystemPermissionType`

- Permission codes should be all lowercase without white spaces
- Specify one permission per action
- Name permission codes as a verb for the entity name and action
- Permission code: [entity][action] – can be interchanged

### 2.5.2 Checking permissions with Gosu expressions

**perm** is a Gosu namespace used to create expressions that determine if current user has a given permission. The expression returns true or false.

- **perm.System.permission** is used when user calls a system permission defined in the `SystemPermissionType` typelist.
- **perm.Entity.permission** is used when an application permission key is used.

### 2.5.3 Static and object-based permissions

An **Application Permission Key** (APK) is a set of one or more system permissions defined in `security-config.xml`.

- APKs use custom handlers to define the mapping of different objects in PolicyCenter to system permissions. For example, `StaticHandlers`, `WrapHandler`, `AccountProducerCodeHandler`, etc.
- PolicyCenter defines APKs for following objects: Account, PolicyPeriod, Jobs, Notes, Documents, Activities, Etc.

**Static** permissions do not require an object as an argument and it may or may not use APKs.

**Object-based** permissions always require an object as an argument and use APKs.

Syntax

- Static System Permissions

***perm.System.permission***

- permission: typecode of a permission defined in `SystemPermissionType` typelist.

- Static Permissions on Entities

***perm.Entity.permission***

- Entity: entity on which the permission is defined.
- permission: permKey attribute defined in security-config.xml.
- it is the APK Defined using StaticHandlers in security-config.xml.
- Object Based Permissions

***perm.Entity.permission (object)***

- Entity: entity on which the permission is defined.
- permission: permKey attribute defined in security-config.xml.
- object: the current object in memory
- It is the APK defined using custom Handlers in security-config.xml.

