

# Configuring Roles Using AXL

A blue square graphic with white curved lines forming a stylized globe or network pattern.

## Cisco Confidential

This document contains valuable trade secrets and confidential information belonging to Cisco Systems, Inc. and its suppliers. The aforementioned shall not be disclosed to any person, organization, or entity, unless such disclosure is subject to the provisions of a written non-disclosure and proprietary rights agreement, or intellectual property license agreement, approved by Cisco Systems, Inc. The distribution of this document does not grant any license or rights, in whole or in part, to its content, the product(s), the technology(ies), or intellectual property, described herein.

## Disclaimer

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS DOCUMENT ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENT ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

## Table of Contents

<b>Introduction .....</b>	<b>3</b>
<b>Roles .....</b>	<b>3</b>
<b>Current Support For Roles .....</b>	<b>4</b>
<b>Database Structure .....</b>	<b>4</b>
<b>Configuring Roles .....</b>	<b>6</b>
<b>Adding New Roles .....</b>	<b>6</b>
<b>Example.....</b>	<b>6</b>
<b>Error Cases: .....</b>	<b>6</b>
<b>Finding Roles.....</b>	<b>6</b>
<b>Example.....</b>	<b>6</b>
<b>Removing Roles .....</b>	<b>7</b>
<b>Example.....</b>	<b>7</b>
<b>Associating Privileges.....</b>	<b>7</b>
<b>Finding Applications .....</b>	<b>7</b>
<b>Example.....</b>	<b>7</b>
<b>Finding Resources Associated To An Application.....</b>	<b>7</b>
<b>Example.....</b>	<b>8</b>
<b>Adding Privileges .....</b>	<b>8</b>
<b>Table : Privilege Details .....</b>	<b>8</b>
<b>Example.....</b>	<b>9</b>
<b>Error Cases: .....</b>	<b>9</b>
<b>Removing Privileges .....</b>	<b>10</b>
<b>Example.....</b>	<b>10</b>
<b>Conclusion .....</b>	<b>10</b>
<b>For More Information .....</b>	<b>10</b>

# Introduction

This document describes how the Cisco Unified Communications Manager (CUCM) user can configure 'Roles' using AXL. It describes the definition of roles and the procedure how to add/update user specific roles and add/update privileges to various resources of each application in an orderly fashion. The tables involved and the inter-relationship between them is also depicted. Sufficient examples have been interspersed to provide a complete understanding of how to handle roles using AXL.

## Roles

A role includes a collection of resources for an application (figure 1), such as the Cisco Unified Communications Manager Administration application. Two types of roles exist: standard roles, which are the default roles, and custom, administrator-defined roles. Standard roles for an application get created upon installation of the application. Administrators may define custom roles.

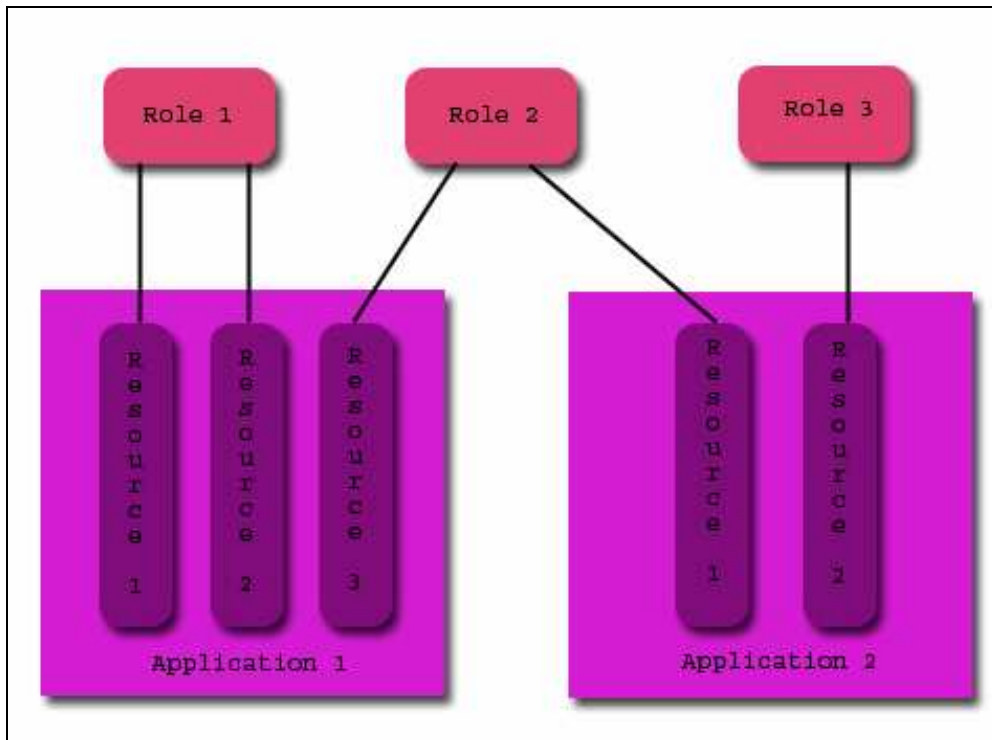


Figure 1: Relation between roles, applications and resources

**Note:** All standard roles get created at installation. You cannot modify or delete standard roles, but you can copy them to create new custom roles based on standard roles.

## Current Support For Roles

CUCM Administration GUI has full support to handle roles.

Currently there is no thick AXL API for roles. However we can use thin AXL APIs `executeSQLUpdate` and `executeSQLQuery` as a workaround method to support roles.

## Database Structure

Before we go ahead to discuss the configuration of roles using AXL it is important to understand the tables involved and the relationship between them. There are 4 main tables involved: `functionrole`, `typeapplication`, `typeresource` and `functionroleresourcemap` (figure 2 and 3).

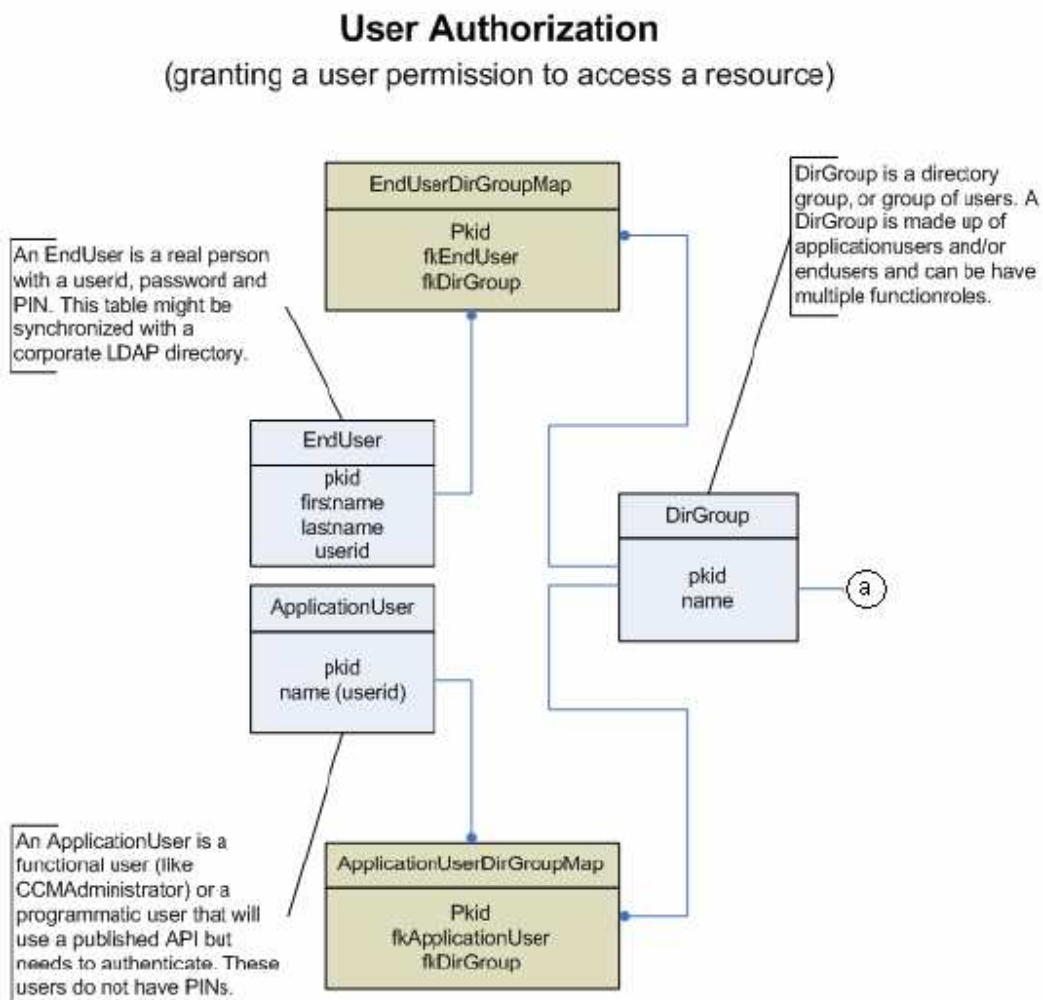
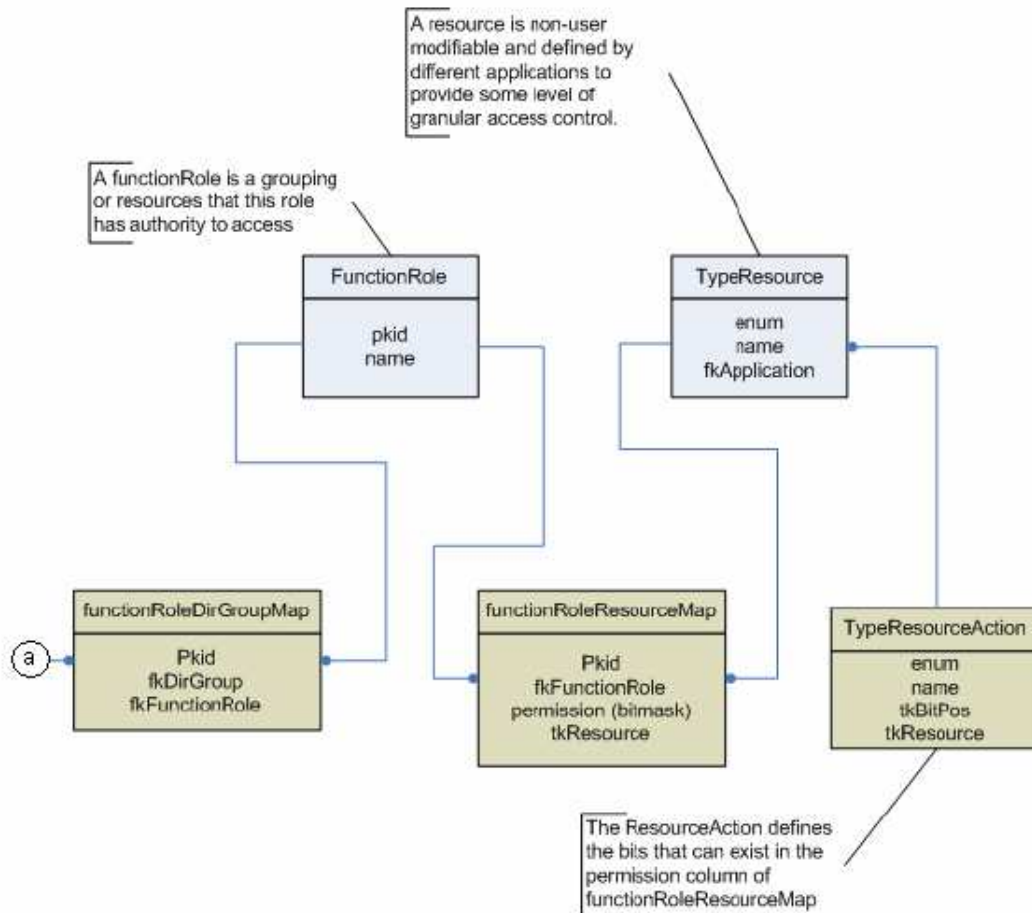


Figure 2: Inter-tabular Relationship Part 1

## User Authorization (granting a user permission to access a resource)



**Figure 3: Inter-tabular Relationship Part 2**

Given below is a brief description of the tables:

- 1) **functionrole**: This table contains the entries for each role. It has 4 fields: `pkid`, `description`, `name` and `isstandard`. The `isstandard` field if has value "true" then it means it is a standard role. Others have `isstandard` as "false".
- 2) **typeapplication**: This table contains the entries for each application. It has 4 fields: `pkid`, `name`, `vroot` and `moniker`. We do not modify this table at any time.
- 3) **typeresource**: This table contains the entries for each resource. It has 5 fields: `enum`, `name`, `moniker`, `tkapplication` and `prefix`. The `tkapplication` field refers to the entry in `typeapplication` table to which the resource is associated. Each resource is associated to only one application.
- 4) **functionroleresourcemap**: This table maps together entries roles and the resources. It has 4 fields: `pkid`, `fkfunctionrole`, `tkresource` and `permission`. The `fkfunctionrole` refers to the role and `tkresource` refers to the resource. The value of `permission` determines what kind of privilege is set for this resource for the given role.

# Configuring Roles

This section describes how to add/update/get/remove roles using AXL.

## Adding New Roles

AXL API executeSQLUpdate can be used to send an insert query to create a new roles. The entry for each role is stored in “functionrole” table.

### Example

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
  <axlapi:executeSQLUpdate sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/1.0"
  xmlns:axl="http://www.cisco.com/AXL/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance" xsi:schemaLocation="http://www.cisco.com/AXL/API/1.0 axlsoap.xsd">
    <sql>insert into functionrole (pkid, description, name, isstandard) values (newId(), 'CTI testing', 'CTI
  Temporary', 'f')</sql>
  </axlapi:executeSQLUpdate></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

### Error Cases:

1) name is mandatory: <sql> insert into functionrole (pkid, description, isstandard) values (newId(), 'CTI testing', 'f')</sql>

ERROR: Cannot insert a null into column (functionrole.name).

2) isstandard can have 't' or 'f' only: <sql>insert into functionrole (name, isstandard) values ('CTI Temporary', 'false')</sql>

ERROR: It is not possible to convert between the specified types.

## Finding Roles

AXL API executeSQLQuery can be used to find existing roles.

### Example

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
  <axlapi:executeSQLQuery sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/1.0"
  xmlns:axl="http://www.cisco.com/AXL/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance" xsi:schemaLocation="http://www.cisco.com/AXL/API/1.0 axlsoap.xsd">
    <sql>select pkid, name from functionrole where name='CTI Temporary'</sql>
  </axlapi:executeSQLQuery></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

## Removing Roles

AXL API executeSQLUpdate can be used to remove unwanted roles. When a role is removed then all associated privileges are also removed.

### Example

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<axlapi:executeSQLUpdate sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/1.0"
xmlns:axl="http://www.cisco.com/AXL/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.cisco.com/AXL/API/1.0 axlsoap.xsd">
    <sql>delete from functionrole where name="CTI Temporary" </sql>
</axlapi:executeSQLUpdate></SOAP-ENV:Body> </SOAP-ENV:Envelope>
```

## Associating Privileges

Roles are application specific. Applications are associated with resources. On a whole roles can be used to define the access privileges of the resources of a particular application. Hence first the application must be chosen.

## Finding Applications

AXL API executeSQLQuery can be used to list out the existing applications.

### Example

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<axlapi:executeSQLQuery sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/1.0"
xmlns:axl="http://www.cisco.com/AXL/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.cisco.com/AXL/API/1.0 axlsoap.xsd">
    <sql>select name as Application , enum as tkapplication from typeapplication</sql>
</axlapi:executeSQLQuery></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

The 'tkapplication' value got is used in the next step.

## Finding Resources Associated To An Application

After the application is chosen we choose the resource associated with that application for which the access privileges are to be defined. AXL API executeSQLQuery can be used to get all resources associated with an application.

## Example

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<axlapi:executeSQLQuery sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/1.0"
xmlns:axl="http://www.cisco.com/AXL/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.cisco.com/AXL/API/1.0 axlsoap.xsd">
    <sql>select name as Resource,enum as tkresource from typeresource where tkapplication=3
ORDER BY name</sql>
</axlapi:executeSQLQuery></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

The 'tkresource' got is used in the next step.

## Adding Privileges

Once we have decided upon the application and its resource then we can define its privilege for a role. Given below is a table which defines the various privileges. Each application has its own set of resources which in turn have their own set of privileges. The privileges also vary for different CUCM versions[5.1,6.0,6.1 etc]. AXL API executeSQLUpdate is used to add privileges using the 'tkresource' from previous step and 'permission' value from the given table.

**Table : Privilege Details**

Application	Privilege	Permission	CM v5.1	CM v6.0	CM v6.1
<b>Cisco Unified Communications Manager Administration</b>	Read	1	Y	Y	Y
	Update	2	Y	Y	Y
<b>Cisco Unified Communications Manager Serviceability</b>	Read	1	Y	Y	Y
	Update	2	Y	Y	Y
<b>Cisco Computer Telephone Interface (CTI)</b>	Allow control of all devices	1	Y	Y	Y
	Enable CTI SRTP key distribution	4	Y	Y	Y
	Enable CTI security	2	Y	Y	Y
	Enabled	1	Y	Y	Y
	Allow retrieval	1	Y	Y	Y
	Allow modification	1	Y	Y	Y
	Allow monitoring	1	N	Y	Y
	Allow recording	1	N	Y	Y
<b>Cisco Unified Communications Manager AXL Database</b>	Allow to use API	1	Y	Y	Y
<b>Cisco Extension Mobility</b>	Allow	1	Y	Y	Y
<b>Cisco Unified Communications Manager End User</b>	Read	1	Y	Y	Y
	Update	2	Y	Y	Y
<b>Cisco Unified Reporting</b>	Read	1	Y	N	Y
	Update	2	Y	N	Y



## Example

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<axlapi:executeSQLUpdate sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/1.0"
xmlns:axl="http://www.cisco.com/AXL/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.cisco.com/AXL/API/1.0 axlsoap.xsd">
    <sql>insert into functionroleresourcecmap (fkfunctionrole, tkresource, pkid, permission) values
('d76dc0a6-435b-454e-a671-b1c00eeebfef', 92, newId(), 3)</sql>
</axlapi:executeSQLUpdate></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Privilege details are stored in 'functionroleresourcecmap' table. For each resource there is a separate entry. The fkfunctionrole value is the pkid got in the step "Finding Roles", tkresource is got in step "Finding Resources Associated To An Application" and permission is got from the table given above.

In case of an application having a resource with multiple privileges, for eg. In the application Cisco Computer Telephone Interface (CTI) the "CTI Application" resource has three privileges: "Enable CTI SRTP key distribution", "Enable CTI Security" and "Enabled". Then the permission values of all those privileges which are to be enabled are added and the sum is given in the SQL query. In the example query given above "Enable CTI Security" and "Enabled" privileges are enabled.

**Note:** In order to find out which privileges are associated with a particular resource the CUCM Administration GUI has to be referred.

## Error Cases:

1) fkfunctionrole must be valid: <sql>insert into functionroleresourcecmap (fkfunctionrole, tkresource, pkid, permission) values ('19c7ad92-8a5e-4bb8-8105-fced2bdd35d6', 92, newId(), 100)</sql>

ERROR: Missing key in referenced table for referential constraint (informix.fk\_functionroleresourcecmap\_fkfunctionrole).

2) tkresource, fkfunctionrole and permission are mandatory fields: <sql>insert into functionroleresourcecmap (fkfunctionrole, permission) values ('19c7ad92-8a5e-4bb8-8105-fced2bdd35d6', 3)</sql>

ERROR: Cannot insert a null into column (functionroleresourcecmap.tkresource).

<sql>insert into functionroleresourcecmap (fkfunctionrole, tkresource) values ('19c7ad92-8a5e-4bb8-8105-fced2bdd35d6', 92)</sql>

ERROR: Cannot insert a null into column (functionroleresourcecmap.permission).

<sql>insert into functionroleresourcecmap (permission, tkresource) values (3, 92)</sql>

ERROR: Cannot insert a null into column (functionroleresourcecmap.fkfunctionrole).

3) tkresource value should be valid: <sql>insert into functionroleresourcecmap (fkfunctionrole, tkresource, pkid, permission) values ('18c7ad92-8a5e-4bb8-8105-fced2bdd35d6', 646, newId(), 3)</sql>

ERROR: Missing key in referenced table for referential constraint (informix.tk\_functionroleresourcecmapTkresource).

4) permission value greater than 7: The remainder fetched when the number is divided by 8 is the value used to decide the privilege. For instance if for CTI Application resource we set the permission as 100. Then the remainder got on dividing 100 by 8 is 4. Which according to the table given above will decide the privilege to be 'Enable CTI SRTP key distribution'.

## Removing Privileges

AXL API executeSQLUpdate can be used to change the privileges. To disable a resource completely the particular entry for that resource is deleted from functionroleresourcemap table.

### Example

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<axlapi:executeSQLUpdate sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/1.0"
xmlns:axl="http://www.cisco.com/AXL/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.cisco.com/AXL/API/1.0 axlsoap.xsd">
    <sql>delete from functionroleresourcemap where fkfunctionrole='d76dc0a6-435b-454e-a671-
b1c00eeebfef' and tkresource=92</sql>
</axlapi:executeSQLUpdate></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

**Note:** A role must have atleast one privilege enabled. In other words for each role in 'functionrole' table there should be atleast one entry in 'functionroleresourcemap' table.

## Conclusion

It is seen thus that to add/update roles and add/update related privileges we use raw SQL queries using thin AXL APIs executeSQLQuery and executeSQLUpdate.

## For More Information

- **Roles :** For further indept information on Roles please refer the Cisco Unified Communications Manager Administration Guide.
- **Programming Guides:** The AXL related programming guides can be found at :  
[http://www.cisco.com/en/US/products/sw/voicesw/ps556/products\\_programming\\_reference\\_guides\\_list.html](http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_reference_guides_list.html)