NexJ Customer Relationship Management

Access Control Lists

Technical Discussion

NexJ Customer Relationship Management Notices

Legal Notice

The NexJ Software identified above including this documentation (the “Software”) contain proprietary information and are provided only under the terms of a NexJ Software License and Maintenance Agreement containing restrictions on use and disclosure and are also protected by Copyright and other intellectual property laws. IF YOU HAVE NOT AGREED TO THE TERMS OF A NEXJ SOFTWARE LICENSE AND MAINTENANCE AGREEMENT YOU ARE NOT PERMITTED TO USE THIS DOCUMENTATION AND MUST RETURN IT IMMEDIATELY. You are not permitted to reverse engineer or convert the Software into human readable form. Except as may be expressly permitted in your Software License and Maintenance Agreement, no part of the Software including this documentation may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

The information contained in this document is subject to change without notice. If you find any problems in this documentation, please report them in writing. This documentation is not warranted to be error free. No warranties or conditions are provided for the Software or this documentation except as expressly set out in your Software License and Maintenance Agreement.

NexJ is a registered trademark of NexJ Systems Inc. Business is about Relationships and other trademarks, trade names and logos of NexJ Systems Inc. are the property of NexJ Systems Inc. Other names may be trademarks of their respective owners.

This publication may contain hyperlinks or references to web sites, products, services or publications of persons or companies other than NexJ Systems Inc. Inclusion of such hyperlinks and references does not imply any endorsement or approval of such web sites, products, services or publication by NexJ Systems Inc. You bear all risks associated with the use of such content. If you choose to purchase products or services from a third party, the relationship is directly between you and the third party. NexJ Systems Inc. disclaims all liability for any loss or damage of any sort that you may incur from dealing with any third party.

10 York Mills Road, Suite 700
Toronto, Ontario M2P 2G4
CANADA

Phone: 1-416-222-5611
Fax: 1-416-222-8623
Web: *www.nexj.com*

Contents

1

Introduction

In this chapter:

Access Control List (ACL) functionality permits and restricts access to objects in NexJ Customer Relationship Management.

The following objects can be secured through ACL:

Entities

Contacts, companies, and households.

Opportunities

Opportunities associated with contacts, companies, and households.

Interactions

Interactions associated with contacts, companies, and households.

An *interaction* is any activity in NexJ CRM, including schedule items, activity plans, tasks, documents, emails, and call records.

Interactions can be single events, such as a schedule item reminding users to call a client, or they can be recurring events, such as a weekly status meeting.

This document provides an overview of how ACL controls access to data through the Group Security Model (GSM), Coverage Access Model (CAM), and Hierarchical Access Model (HAM), describes configuration and deployment options, and describes ACL security classes and security rules.

Overview

ACL functionality controls access to data in NexJ CRM. An *Access Control List* is a table containing a list of objects and the principals that have access to the objects. A *principal* is a user or user group defined by your security model.

All users belong to the Public group. They can also belong to one or more of the following:

- A private group in which they are the only member.
- A group that contains other members.
- A group automatically populated based on an organizational hierarchy.

ACL tables are populated depending on the security model deployed in your project. Separate tables are created for each object that is secured by ACL.

Access Control List (ACL) functionality was introduced in NexJ Contact for Finance and Insurance in version 7.X and enhanced in NexJ CRM in version 8.X for easier customization. Previously, security was implemented on objects in NexJ CRM using the legacy Group Security Model (GSM). Legacy GSM enables you to secure an object to a single principal. ACL replaces the legacy model and enables you to secure an object to multiple principals. The GSM now implements ACL for contacts, companies, households and their related opportunities and interactions. Legacy GSM security continues to apply to other application objects in NexJ CRM, such as campaigns and service requests.

ACL introduces two new security models, the Coverage Access Model (CAM) and the Hierarchical Access Model (HAM). The CAM uses a peer-to-peer relationship by granting access to objects through membership in a coverage group. The HAM uses a top-down approach to security, where users higher in the hierarchy have access to objects associated with users lower in the hierarchy.

Security models

Security models allow explicit or implicit access to objects. ACL can allow explicit or implicit access to an object, in the following ways:

Explicit access

Access is given to specific users who require access to an object, regardless of other security rules. For example, users invited to a meeting must have access to the meeting even if they are not part of the appropriate security group.

Implicit access

Access is given to users through associative access or hierarchical access. *Associative access* is given to users who require access based on their role, for example, to a user who covers contacts for another advisor who is on vacation. *Hierarchical access* is given to users based on the position in the company, for example, to a manager who requires access to the advisors who report to them.

The Group Security Model applies explicit security based on direct user and group assignment. The Coverage Access Model and Hierarchical Access Model apply implicit security based on a object's coverage team or on an organizational hierarchy.

Group Security Model

The Group Security Model enables users to apply public, group, and private view and edit security settings to contacts, companies, households, and their related opportunities and interactions in NexJ CRM. Security settings are applied to individual objects using the **Security** dialog. *Figure 1: Contact Security dialog* on page 3 shows view and edit security settings in the **Security** dialog for a contact.

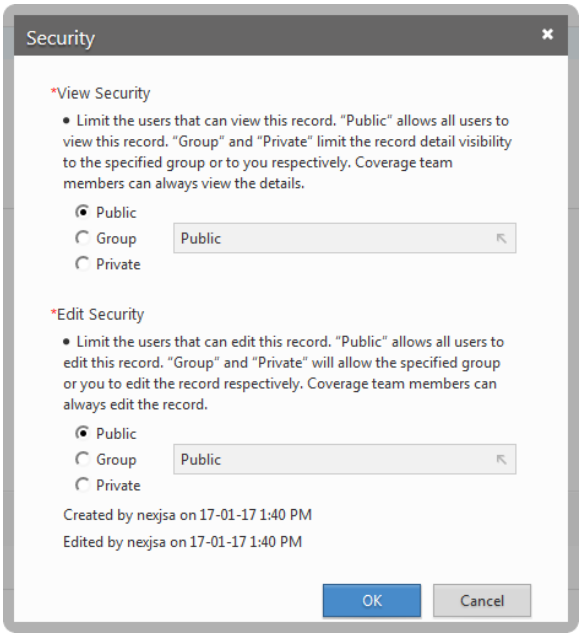

Figure 1: Contact Security dialog

For example, you can use ACL functionality to mark an interaction as private.

The Group Security Model is disabled when the Hierarchical Access Model is deployed.

Coverage Access Model

The Coverage Access Model restricts data access in NexJ CRM by coverage team. Users receive view and edit access to an entity and its related opportunities and interactions by being a member of the coverage team for a contact, company, or household.

For example, you can use ACL functionality to:

- Secure a contact using a coverage team.
- Enable an opportunity or interaction to inherit security from a coverage team.

The Coverage Access Model is always enabled.

Hierarchical Access Model

The Hierarchical Access Model restricts data access in NexJ CRM by organizational hierarchy. A user's position in the hierarchy determines which data they can access. Users higher in the hierarchy have view or edit access to data of users lower in the hierarchy. The Hierarchical Access Model applies to contacts, companies, households, and their related opportunities and interactions.

For example, you can use ACL functionality to:

- Secure a contact using a rep code. A *rep code* is an identifier assigned to an employee such as a registered representative.
- Enable a manager to see the contacts, opportunities, and interactions that his direct reports can see.
- Enable an opportunity to inherit security from contacts in the opportunity's **For** field.
- Enable an interaction to inherit security from contacts in the interaction's **For** field and users in the **Assigned To** field.

Single and multiple inheritance

ACL propagation occurs through single or multiple inheritance in the Coverage Access Model and Hierarchical Access Model.

An object can inherit security through single or multiple inheritance.

Single inheritance

Figure 2: Single inheritance on page 4 shows how an interaction or opportunity inherits security from an entity.

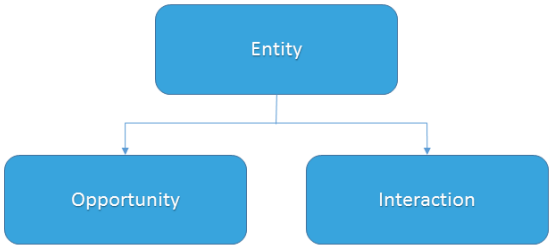

Figure 2: Single inheritance

Multiple inheritance

An interaction or opportunity can also inherit security from multiple parents. *Figure 3: Multiple inheritance* on page 4 shows an interaction inheriting security in the following ways:

- The opportunity inherits security from the entity and the interaction inherits security from the opportunity.
- The interaction implicitly inherits security from the entity.

Additional constraints can be placed on the child interaction and opportunity, so that the users who can access the children is a subset of the users who can access the parent entity.

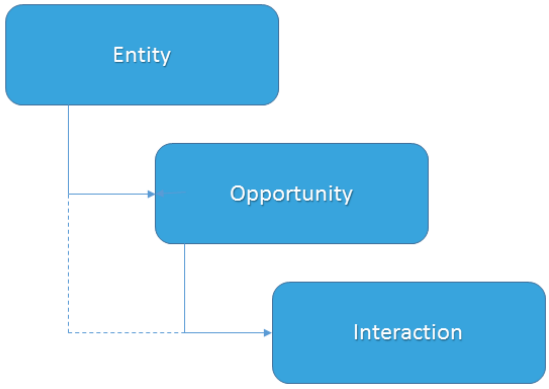

Figure 3: Multiple inheritance

2

Enabling security models

In this chapter:

This chapter provides an example of how to enable HAM and define an organization hierarchy, enable CAM for a user, and implement CAM for the entity class.

The section also describes the default implementation of the GSM, CAM, and HAM security models.

Enabling the Hierarchical Access Model (HAM)

You enable HAM for your deployment by:

1. Customizing the `SystemPreference` class in NexJ Studio.
2. Enabling the `SystemPreference` attribute for HAM in the seed event for the class.
3. Reseeding the database.

Note: When HAM is enabled, default ACL rules will begin executing. You must ensure that you disable rules that you do not want apply to your deployment by first customizing the `ACL_SECURABLE_CUSTOM` library.

After deploying your project, HAM becomes enabled and the Group Security Model is disabled. In NexJ Admin Console, in the **Global Application Settings** page, the Hierarchical Access feature displays as enabled, as shown in *Figure 4: Global Application Settings page* on page 6.

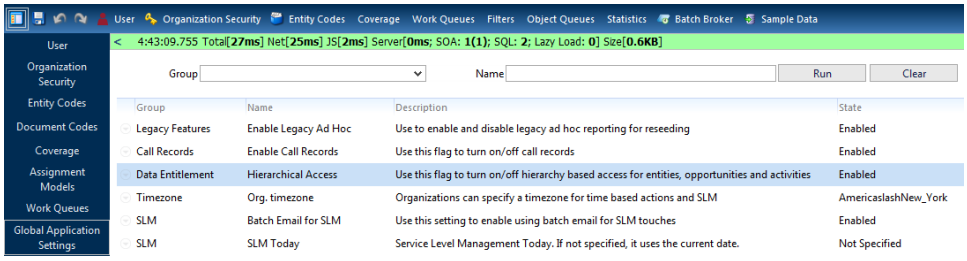

Figure 4: Global Application Settings page

When HAM is enabled, **Security** dialogs and tabs are disabled for contacts, companies, households, and their related opportunities and activities in NexJ CRM. For other application objects, the **Security** tab continues to be available.

Figure 5: Security menu option enabled on page 6 shows the **Security** menu option for a contact, before HAM is enabled.

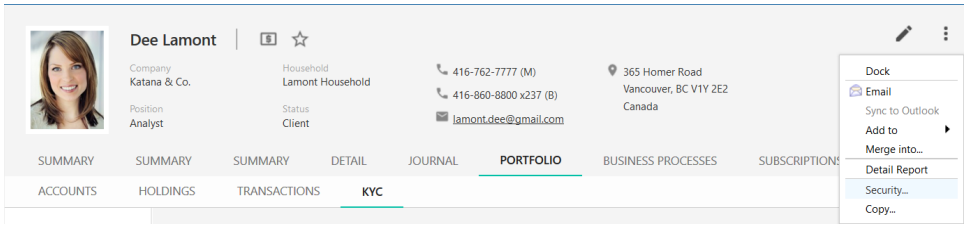

Figure 5: Security menu option enabled

With HAM enabled, the **Security** menu option is disabled, as shown in *Figure 6: Security menu option disabled* on page 6.

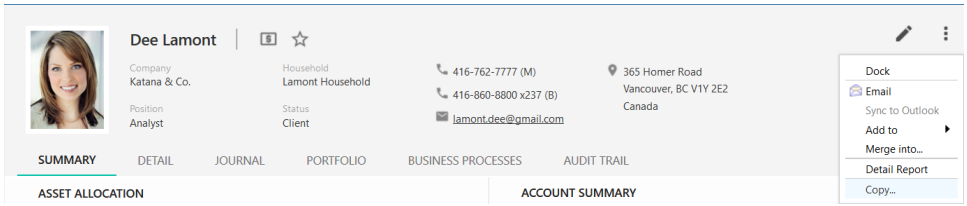

Figure 6: Security menu option disabled

Before HAM is enabled, the **Security** tab is also enabled in schedule items for contacts, as shown in *Figure 7: Security tab enabled* on page 7.

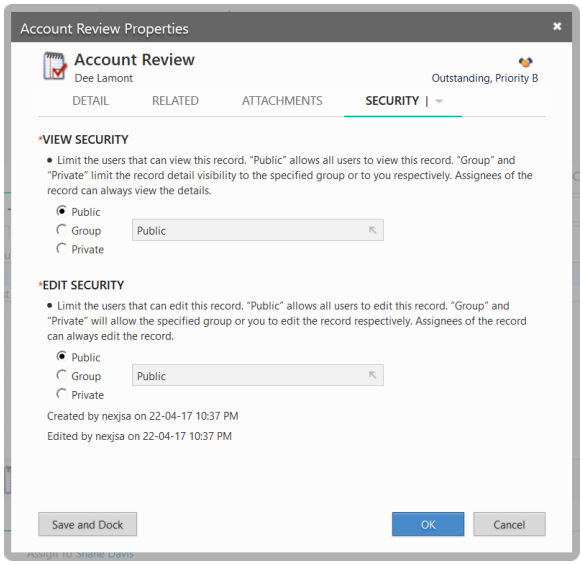

Figure 7: Security tab enabled

With HAM enabled, the **Security** tab is disabled, as shown in *Figure 8: Security tab disabled* on page 7.

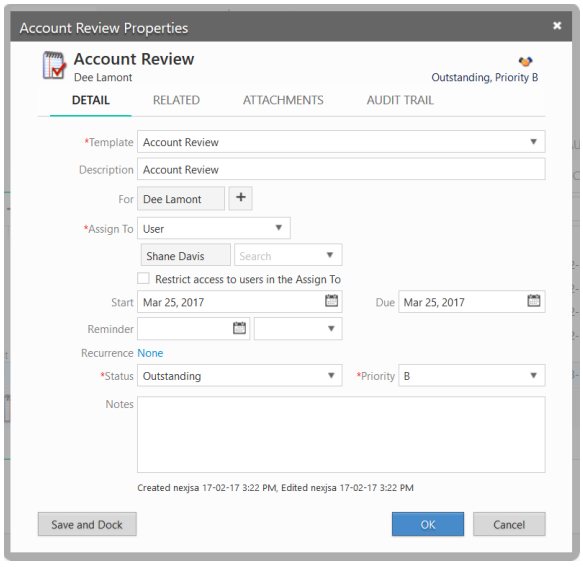

Figure 8: Security tab disabled

With HAM enabled, the **Rep Code** field is available in the **Add** and **Edit** dialogs for contacts, companies, and households. *Figure 9: Rep Code field enabled* on page 8 shows the **Rep Code** field in the **Edit** dialog for a contact.

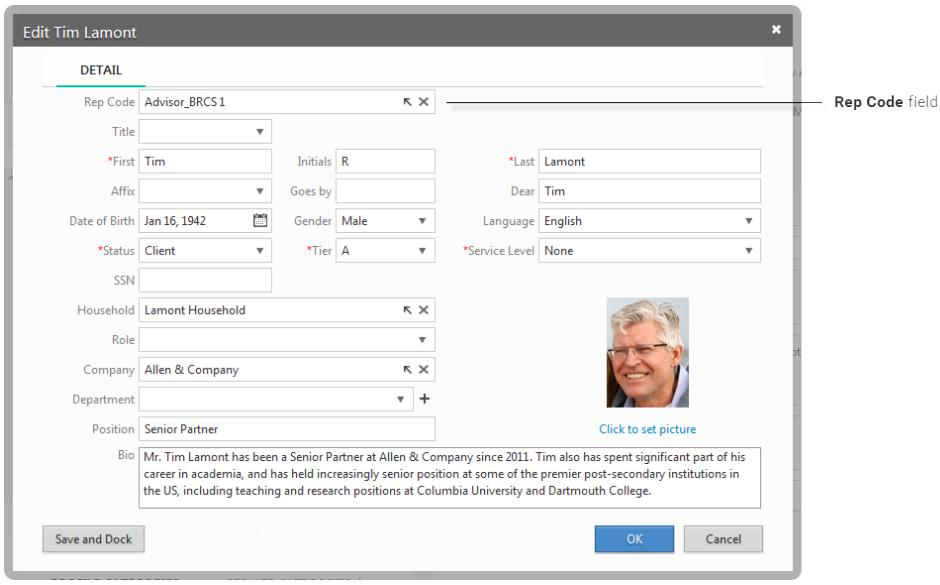

Figure 9: Rep Code field enabled

Defining an organizational hierarchy

You define your organizational hierarchy on the **Organization Security** page in NexJ Admin Console.

The hierarchy contains four levels: Rep Code, Branch, Division, and SubFirm. *Figure 10: Organization Security page* on page 8 shows the **RepCode** tab selected on the **Organization Security** page.

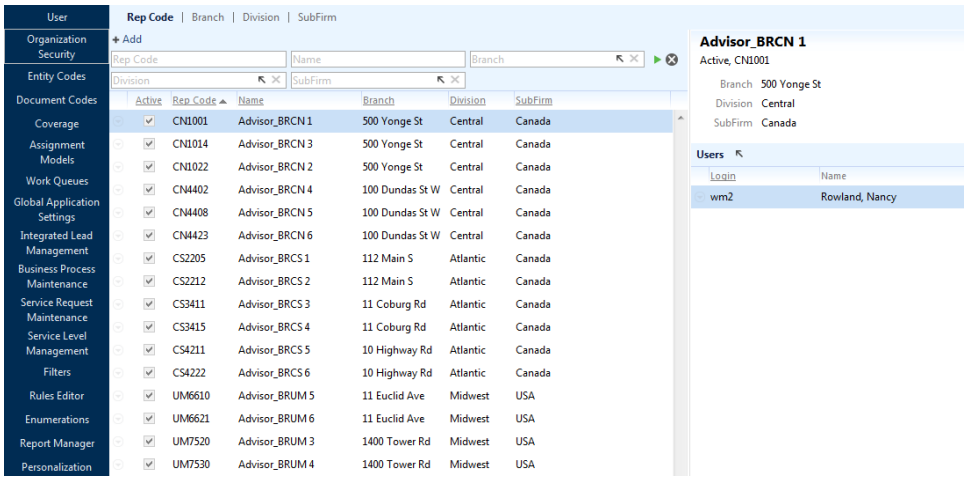

Figure 10: Organization Security page

Typically, the Rep Code level applies to financial advisors and sales representatives, the Branch level applies to branch managers, and Divisions and SubFirms apply to executives. A branch could represent a physical location, a division a larger geographic sales region, and a subfirm a country.

For example, the rep code for a financial advisor is assigned to the contacts that belong to a financial advisor. The financial advisor can see those contacts and their opportunities and activities. A branch manager can view the contacts that belong to the advisors in their branch. The branch manager can also see opportunities and activities for those contacts. Executives who are responsible for a division can see contacts, opportunities, and activities belonging to the branch managers and their advisors in their sales region. Finally, executives responsible for a larger region, such as a country, can see the contacts, opportunities, and activities belonging to those divisions.

Figure 11: ACL hierarchy on page 9 shows a simple hierarchy.

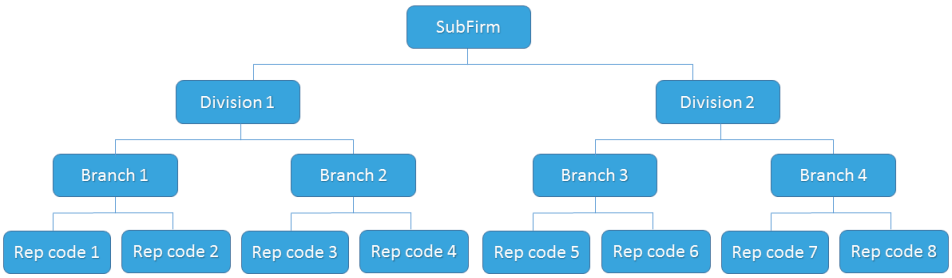

Figure 11: ACL hierarchy

Individual users are assigned to the hierarchy on the **User** page in NexJ Admin Console. An administrator selects the rep codes, branches, divisions, or subfirms to assign in the **Data Entitlements** tab for a user. Rep codes are assigned to individual contacts on the **Contacts** workspace in NexJ CRM. *Figure 12: Data Entitlements tab* on page 9 shows the **Data Entitlements** tab in NexJ Admin Console.

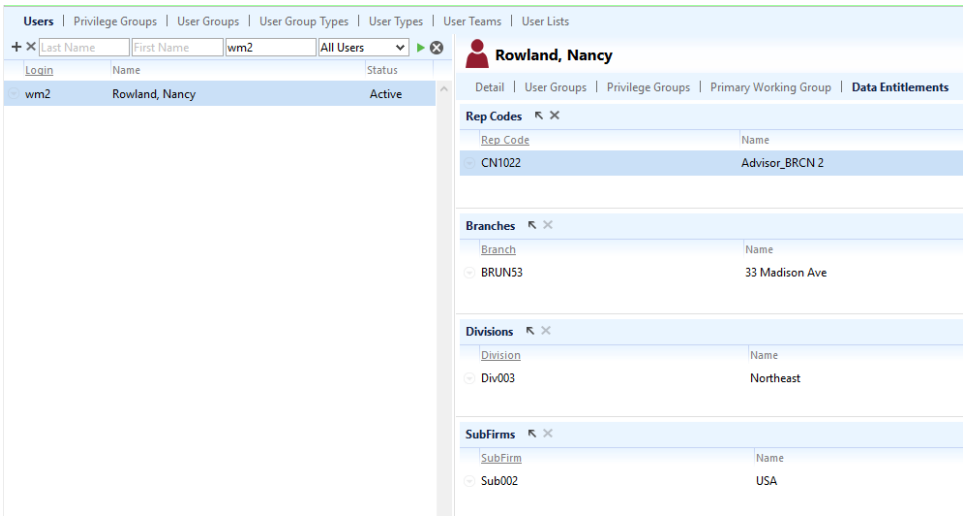

Figure 12: Data Entitlements tab

Note: A contact, company, or household can be accessed by all users if they are not assigned a rep code.

For more information on configuring the hierarchy in NexJ CRM, see *NexJ Customer Relationship Management Administrator Guide*. For more information on assigning rep codes to contacts, see *NexJ Customer Relationship Management User Guide*.

Implementation of HAM on an entity

This example shows how hierarchy access is implemented on the entity class.

When a entity, such as a contact, is assigned a rep code, an `EntityRepCode` instance is created for the contact. The `EntityRepCode` instance associates the contact with the hierarchy which grants users access to the contact. Users are entitled to data depending on the levels of the hierarchy that they are assigned to, as shown in *Figure 13: Data entitlement by hierarchy level* on page 9.

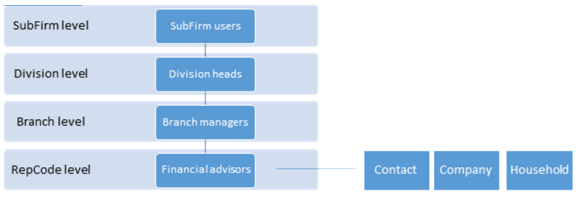

Figure 13: Data entitlement by hierarchy level

Enabling access to a contact using the Coverage Access Model (CAM)

In the Coverage Access Model, users are provided with access to an entity by being a member of the coverage team for a contact, company, or household. Users also have access to the opportunities and activities that are associated with the entity.

Figure 14: Adding user to coverage team on page 10 shows a user being added to the coverage team for a contact. The user will have view and edit access to the contact and his opportunities and activities.

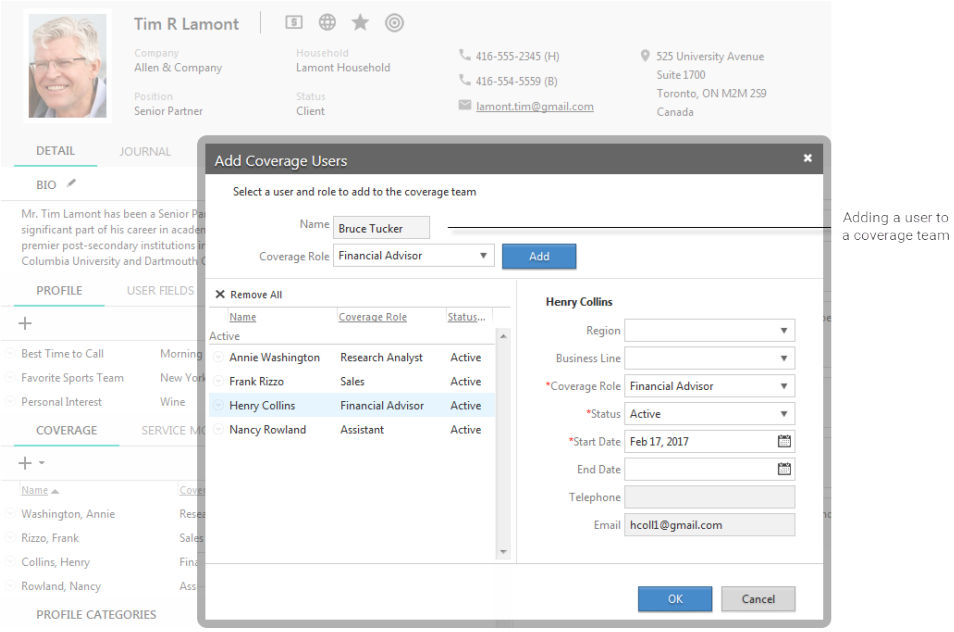

Figure 14: Adding user to coverage team

Implementation of CAM on an entity

This example shows how entity coverage is implemented on the entity class.

Figure 15: Implementation of the ACL_SECURABLE aspect on page 10 shows the relationship between the `ACL_SECURABLE` aspect, `Entity` class, `EntityCoverage` class, and `EntityACLEntry` class.

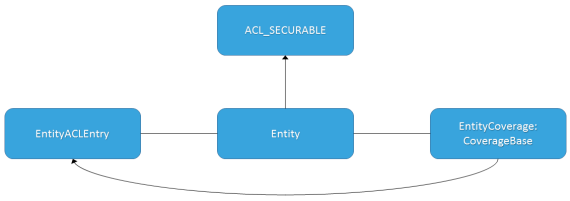

Figure 15: Implementation of the ACL_SECURABLE aspect

The `ACL_SECURABLE` aspect is applied to the `Entity` class. The `ACL_SECURABLE` aspect calls functions that use ACL logic to populate the `EntityACLEntry` table. For example, functions apply read and edit access to entities, get owners based on security rules, and recalculate the `EntityACLEntry` table based on additions, deletions, or changes to entity owners.

The `EntityCoverage` class triggers ACL rebuild logic when coverage changes for an entity. The `EntityACLEntry` class persists links between entities with the users who can view or edit them in the `EntityACLEntry` table. When a user is assigned coverage of a contact, principal information for the user is added to the `EntityACLEntry` table.

For example, an ACL rebuild is triggered when a user is assigned coverage of a contact through a coverage team. Principal information for the user is added to the `EntityACLEntry` table.

Default Group Security Model implementation

The Group Security Model applies to contacts, companies, households and their related opportunities and interactions.

The following security settings are available when using the Group Security Model:

- **Public** enables all users with access to the record.
- **Group** enables users in the specified group with access to the record.
- **Private** enables the record owner with access to the record. Also enables access for record owners and coverage team members to a contact or opportunity, and users specified in the **Assign To** field in an activity.

This rule is implemented by the `ActACLEntry`, `ActionItemSeriesACLEntry`, `BaseOpportunityACLEntry`, `EntityACLEntry`, and `RoadshowACLEntry` classes.

Default Coverage Access Mode implementation

The Coverage Access Mode applies to contacts, companies, households and their related opportunities and interactions.

Security is specified for entities and their related opportunities and interactions through the Coverage Access Model in the following ways:

- Adding a user to the coverage group for an entity creates an `EntityACLEntry` record for the user. This gives the user access to the entity and to opportunities and interactions associated with the entity.
- Adding a user to the **Assigned To** field in an interaction creates an `ActACLEntry` or `ActionItemSeriesACLEntry` record for the user. This is implemented through the `ActACLEntry` and `ActionItemSeriesACLEntry` classes.
- Adding users to the **Assigned To** field in an opportunity inherits the entity's security onto the opportunity. This rule is maintained by the `BaseOpportunityACLEntry` class.

Default Hierarchical Access Model implementation

The Hierarchical Access Model applies to contacts, companies, households and their related opportunities and interactions.

Security is specified for entities and their related opportunities and interactions using rep codes in the following ways:

- Assigning a rep code to an entity creates a set of ACL records. An `EntityACL` record is created for each user who has access to the rep code and entity. Users have access through ownership of the rep code and entity or through the hierarchy. This is implemented through the `EntityACL` class.
- Users who have access to the rep code and entity are permitted access to the opportunities and interactions associated with the entity. A `BaseOpportunityACL`, `ActACLEntry`, and `ActionItemSeriesACLEntry` record is created for each user who has access to opportunities and interactions through the rep code and entity. This is implemented through the `BaseOpportunityACL`, `ActACLEntry`, and `ActionItemSeriesACLEntry` classes.
- Adding a user to the **Assigned To** field in an interaction also creates an `ActACLEntry` or `ActionItemSeriesACLEntry` record for the user.

3

Deployment and migration

In this chapter:

This section shows how to modify default security rules and rebuild ACL tables.

Modifying default security rules

The `ACL_SECURABLE` library file contains helper functions that implement default ACL rules. The library contains generic ACL population logic in the `acl:default-get-acl-principals-act` file.

To modify security rules, you customize the `ACL_SECURABLE_CUSTOM` library file. The `ACL_SECURABLE_CUSTOM` library file references functions in the `ACL_SECURABLE` library and contains customizations to the file.

For example, you can modify the `acl:get-default-security-rule`, which contains a mapping used to retrieve the default security rule for a given object. The default security rule mapping shown in the code snippet below specifies that the default security is:

- Private for entities and file imports
- Public for saved lists and system defined queries
- Defined by the Coverage Access Model for events

```
; Key is the SecurityRuleEnum and values are classes that initialize to those
 security rules
; "EntityList", "BatchFileImport" and "FileImport" use "OWNER" as their default
 (private to the creator)
; "SavedFilterRule" is public at creation

(define mapping
   `(
      (,(SecurityRuleEnum'OWNER) EntityList BatchFileImport FileImport)
      (,(SecurityRuleEnum'COVERAGE) roadshow:Roadshow)
      (,(SecurityRuleEnum'PUBLIC) SavedFilterRule)
    )
)
```

Rebuilding ACL tables

After making changes to security rules in your project, you must rebuild ACL tables that are affected by the changes. You typically make changes after initially populating tables, performing a data migration, or after making extensive model changes.

How to rebuild ACL tables

You rebuild ACL tables using the **BatchACLRebuildCommand** batch process in the **Batch Broker** in NexJ Admin Console. When you run the batch process, you can specify arguments to restrict the number of tables that you rebuild. For example, if changes in your project only affect opportunities, you can specify arguments to only rebuild the BaseOpportunityACLEntry table. *Figure 16: Batch Broker dialog* on page 15 shows the sample arguments in the batch broker.

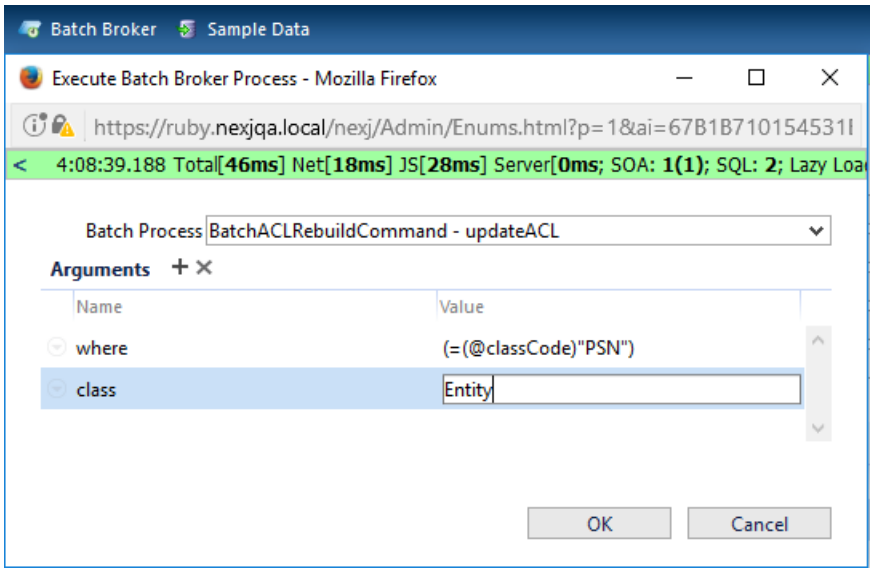

Figure 16: Batch Broker dialog

It is recommended to rewrite the **BatchACLRebuildCommand** batch process in SQL for performance purposes.

Appendix A: Class descriptions

This section contains descriptions of several key ACL classes and aspects. The `BaseACLEntry` class, `ACL_MEMBER` aspect, `ACL_SECURABLE` aspect, and the `SystemPreference` class are described below.

BaseACLEntry

The `BaseACLEntry` class contains links between an `ACL_SECURABLE` aspect and a principal. The `BaseACLEntry` class is an abstract class and has no persistence.

ACL_MEMBER aspect

The `ACL_MEMBER` aspect is applied on classes associated with classes that are secured by ACL. The aspect triggers rebuild logic on the main class. For example, if the `ACL_MEMBER` aspect is applied to the `EntityCoverage` class, the entity's ACL is updated when coverage is applied to an entity in NexJ CRM.

ACL_SECURABLE aspect

The `ACL_SECURABLE` aspect is applied to all classes that are secured by ACL. Within the rebuildACL event, the aspect calls functions defined in the `ACL_SECURABLE` and `ACL_SECURABLE_CUSTOM` libraries, which contain ACL population logic. The ACL_SECURABLE library contains default ACL logic. The `ACL_SECURABLE_CUSTOM` library references functions in the ACL_SECURABLE library and contains customizations that you can create to override the default logic.

When the rebuildACL event occurs for an object, the following occurs:

- ACL entries that no longer valid are deleted.
- ACL entries are rebuilt for the object. Principals are added to the object's ACL table and read-only access is assigned to specific principals.
- ACL records are refreshed for child objects. For example, when an entity is modified, ACL records are also modified for children of the entity, such as opportunities and activities.

SystemPreference

The `SystemPreference` class seeds the `HIERARCHICAL_SECURITY_ENABLED` preference to the setting defined in the environment file. The preference displays as a read-only feature in the **Global Application Settings** page in NexJ Admin Console.

The class also contains the `isCustomSecurityEnabled(metaclass class)` event which returns true for any class having the ACL_SECURABLE aspect and for which the mapping value returns true. The mapping specifies that custom security is always enabled for `EntityList`, `roadshow:Roadshow`, `roadshow:RoadshowScheduleItem`, and `SavedFilterRule` classes. Custom security is enabled through system preferences for the `Act`, `ActionItemSeries`, `Entity`, and `Object` classes.

Appendix B: BaseACLEntry derived classes

`BaseACLEntry` is an abstract class and has no persistence. The following are derived classes which have persisted attributes.

Note: All other objects use the legacy GSM. Discussion of classes that use legacy GSM is beyond the scope of this document.

Class	Description
ActACLEntry	Links interactions with the users who are entitled to view or edit them. Users are entitled to data if they are secured to a contact in the interaction's **For** field or are listed in the interaction's **Assigned To** field.
ActionItemSeriesACLEntry	Links interactions that occur in a series, such as recurring meetings, with the users who are entitled to view or edit them. Users are entitled to data if they are secured to a contact who is listed in the action item's **For** field or are listed in the action item's **Assigned To** field.
BaseOpportunityACLEntry	Links opportunities with the users who are entitled to view or edit them. Users are entitled to data through membership in the opportunity's coverage team or in the hierarchy.
EntityACLEntry	Links contacts, companies, and households with the users who are entitled to view or edit them. Users are entitled to data through membership in the contact's coverage team or in the hierarchy. The `EntityACLEntry` class derives properties and attributes from the `BaseACLEntry` metaclass. **Note:** The `EntityACLEntry` class persists the classCode of the Principal.
EntityListACLEntry	Links saved lists with the users who are entitled to view or edit them.
RoadshowACLEntry	Links events with the users who are entitled to view or edit them.
SavedFilterRuleACLEntry	Links system defined queries with the users who are entitled to view or edit them.

Each derived class persists the following attributes:

objectId

ID of the object with security. Specifies the foreign key to the table which stores the rest of the object data, for example, FK to `Act`.

principalId

ID of the principal that has access to the object. Specifies the foreign key to the related Principal table.

allowEdit

Boolean value which indicates whether the principal has the ability to edit. The value of `1` specifies allow edit and `0` specifies read only.

NexJ Systems is a leading provider of enterprise private cloud software, delivering CRM solutions primarily to the financial services, insurance, and healthcare industries. Our next-generation, people-centred software combines industry-specific functionality with information from multiple applications and data stores to provide comprehensive knowledge of the customer. Organizations use this knowledge to provide superior sales and service by enabling proactive interactions that influence behaviour.

NEXJ SYSTEMS®