Upgrading Registry to v4.0.0

Enrollment Flow Deprecations

As part of the transition to Registry v5.0.0, certain Enrollment Flow configurations have been deprecated. While no changes are currently required, administrators may wish to review Migrating to EnvSource to prepare for future upgrades.

Changes to Enrollment Flow Plugins

As of Registry v4.0.0, Enrollment Flow Plugins are *Instantiated*. There are two major impacts of this change:

1. Plugins will need to support a configuration page, as described here.
2. Plugins are no longer globally enabled for all Enrollment Flows. Instead, Plugins must be attached to each Enrollment Flow on which they are intended to operate. An Enrollment Flow Plugin attached to an Enrollment Flow is called an *Enrollment Flow Wedge*, and is described here.

Identifier Enroller Plugin Configuration

As Enrollment Flow Plugins are now instantiated, the IdentifierEnroller Plugin is configured differently. The Plugin must be attached to the desired Enrollment Flow(s) as an Enrollment Flow Wedge, and then the desired identifiers configured for that flow.

⚠️ Prior configurations will *not* be automatically migrated, since there is not a straightforward way to know which configurations were intended to be used with the Identifier Enroller.

COmanage CO Localizations

Localizations made to the COmanage CO now apply to the entire Platform, unless overridden within another CO.

Attribute Enumerations

Attribute Enumerations are now based on Dictionaries. Dictionaries will automatically be created for existing Attribute Enumerations. No specific manual tasks are required as part of this migration, however deployments with multiple enumerations over the same values may wish to update their configurations to use shared Dictionaries (to avoid duplicate data sets).

⚠️ The AttributeEnumeration API has been removed.

New Organizations Object

Registry v4.0.0 introduces a new Organization object. This isn't of itself an issue regarding attention, except that there was a legacy (never functional) Organization object that was removed in v2.0.0. Deployments that came online on versions prior to v2.0.0 might still have the old cm_organizations (obsolete) table present if it were not manually dropped. If the deployment is also using Postgres, then a schema management bug may prevent the table from being fully converted to the new cm_organizations structure. This will result in some columns having the wrong widths. Operationally, this is unlikely to cause problems, and will eventually get corrected when the underlying schema management bug is addressed. However, affected deployments may wish to manually drop the old table before upgrading, or review the column widths after upgrading to make sure they are correct.

Changes to Job Plugins and Registry Job Shell

Registry v4.0.0 updates the naming conventions for Job Plugins in order to permit polymorphic Plugins, and for a given Plugin to implement more than one Job. In addition, Job Shell native tasks have been moved to the Core Job Plugin. There are several impacts from this change:

1. Cron jobs invoking the Job Shell to run tasks (expirations, groupvalidity, syncorgsources) must be updated to use the new jobs made available via the Core Job Plugin.
2. Custom Job Plugins will need to be updated to the new interface.
3. The ***job*** argument to Registry Job Shell must be updated to use the new naming convention (eg: `FooJob.Bar`).
4. The Identifier Assignment Job Plugin and Provisioner Job Plugin have been consolidated into the new Core Job Plugin.
    a. Identifier Assignment and Provisioner jobs queued prior to upgrading may need to be requeued.

⚠ Org Identity Source plugins that support changelist mode in order to optimize synchronization may incorrectly determine which records have changed following upgrading. The most likely scenario is just that a much longer sync may take place for the first time following upgrading, however to ensure a smooth transition the following SQL should be manually run:

```
SQL> update cm_co_jobs set job_type='CoreJob.Sync' where job_type='OS';
```

Changes to Authenticator and Provisioner Plugins

Several minor changes have been made to Authenticator Plugins and Provisioner Plugins to make their behavior more consistent. In some cases, these changes make the behavior consistent with the documentation, but do so in a backwards incompatible manner. Specifically:

1. Provisioning data is marshaled from Authenticator Plugins using the `current()` interface, rather than manually `find()`ing data. A default implementation is available, though some plugins may need to override this.
2. By default, locked Authenticator records will no longer be provided to Provisioner Plugins. This was actually the documented behavior, but was not consistently implemented.
3. Provisioner Plugins will no longer receive empty arrays for Authenticator models that have no records. (The relevant key will be unset.)
4. Authenticator Plugins that override `SAMController::manage()` must manually call `notify()` to trigger notification on Authenticator update.

Enforcement of CO Person Status During Provisioning

The provisioning behavior had been documented here, however it was not enforced by ProvisionerBehavior and its implementation within Provisioner Plugins was inconsistent. (Only LdapProvisioner really behaved correctly.) Registry v4.0.0 properly enforces the documented behavior. Plugins and deployments that were relying on the old, incorrect behavior may need to be updated appropriately.

Clarification of LdapProvisioner Statuses

Registry v4.0.0 clarifies handling of certain attributes for certain CO Person statuses. Specifically

- `eduPersonEntitlement` is no longer provisioned for statuses other than `Active` or `Grace Period`. This was already true for certain CO Group-based Entitilements.
- Unix Cluster Account provisioning has been made more consistent. See the documentation for details.

Updates to SqlProvisioner Schemas

Registry v4.0.0 adds support for identifying which Org Identity Source provisioned attributes were sourced from. The Target Database Schema must be updated *after* installing Registry v4.0.0 but *before* the provisioner is used. To do so, view the SqlProvisioner Configuration (*Configuration* > *Provisioning Targets* > (Provisioner) > *Configure*), then click *Reapply Target Database Schema*. Once that process completes, click *Resync Reference Data* on the same page. See the documentation for details.

Match Server Attributes

Match Server Attributes must now be explicitly configured. Deployments using Match Servers connected to Pipelines should update their Match Server configuration. Some attributes may need their COmanage Match API Name updated.

Plugin Search Interface

The plugin interface for global search has changed, with a new `$limit` parameter. Existing plugins implementing the `search()` interface will need to be updated.

Enrollment Flow Terms & Conditions

Presentation of Terms & Conditions during an Enrollment Flow has moved to a new step after email confirmation. This allows for T&C to be required for Invitation flows. There are no configuration changes required, however deployers may wish to be aware of the change to the user experience.

Custom Themes and Plugin Theming

MDL library replaced with Bootstrap 4

The Material Design Light library that shipped with previous versions or Registry has been replaced with Bootstrap 4. As of Registry 4.0, the MDL library has been completely removed from the code base. If you made explicit use of this library in a plugin view, you may need to update your code.

Registry 4.0 makes use of Bootstrap for a number of form layout purposes. Bootstrap has been included as boostrap-bundle, and has Popper.js included. You may make use of Bootstrap for custom Plugin theming.

Changes to Custom Themes

Registry 4.0 has a few stylistic changes that may impact a custom theme:

- <footer> is no longer aligned right by default. If you depended on this, you can add the following to your custom theme's CSS:

  Align custom footer

  ```
  #customFooter {
    text-align: right;
  }
  ```

Plugin View Considerations

Table styles: The default styling for tables has changed from "table-layout: auto" (the browser default) to "table-layout: fixed". This has the advantage of rendering tables more quickly and avoiding rendering issues for large text displayed in a table cell. With this setting, tables will render based on the width of the first cell found in each column, typically the table heading (<th>) cells. If no widths are defined, the columns will render with even widths. The change will expose any mismatch with a colspan that doesn't account for the number of cells in the table, so if you find a plugin's tables rendering oddly, check first for a mismatch between the number of cells and a colspan set on the table.

Two classes have been added to improve layout rendering of index tables: if your action buttons on index rows are two or less, add the class *.thinActionButtonsCol* to the table header to improve the column width, e.g. <*th class="thinActionButtonsCol">.* If you have an ordering column, add the class *.order* to the table header to make it thin, e.g. <*th class="order">.* If you would like to override this behavior entirely, set your tables to "table-layout: auto".