Provisioning Broker and Engine

Fundamentals of IdentityIQ Implementation

IdentityIQ 7.0

SailPoint

Overview

Provisioning Broker and Engine

- Provisioning overview
  - What causes provisioning to occur?
  - Provisioning channels
  - Integration Modules and provisioning
- Provisioning architecture
- Provisioning process and components
- Walkthrough of provisioning example
- Debugging tools

What Causes Provisioning?

- User Driven Request (Lifecycle Manager)
  - Identity
  - Entitlement
  - Account
  - Role
- Event Driven (Identity Refresh or Lifecycle Event)
  - Automatic Role Assignment based on Identity Attribute
  - Sunrise/Sunset of Role
  - Joiner/Leaver/Mover/Native Change
- Remediation
  - Certification Item Revoke
  - Policy Violation Remediation

Provisioning Channels

3 Options

- Read/Write Connectors
  - Used to directly connect with many types of target applications
  - Used by some Integration Modules to indirectly connect with target applications
- Integration Configurations
  - Used by some Integration Modules to indirectly connect with target applications
    - Provisioning Integration Modules (PIM): Tivoli Identity Manager
    - Service Desk Integration Modules (SIM): Remedy, ServiceNow, HP Service Manager
- Internal Provisioning (WorkItems/Notifications)
  - Used when no alternative provisioning pathway is defined

Integration Modules

Provisioning Purpose

- Provisioning Integration Modules (PIM)
  - Integrates with 3rd party provisioning providers
  - Provisioning provider handles provisioning requests submitted by IdentityIQ
- Service Desk Integration Modules (SIM)
  - Integrates with support providers to automate ticketing/tracking
  - Service desk personnel responds to ticket to support provisioning
- Mobile Device Management Integration Module (MIM)
  - Integrates with Mobile Device Management (MDM) Systems to indirectly manage mobile devices
  - MDM system performs the specified action on the target device

Provisioning Architecture

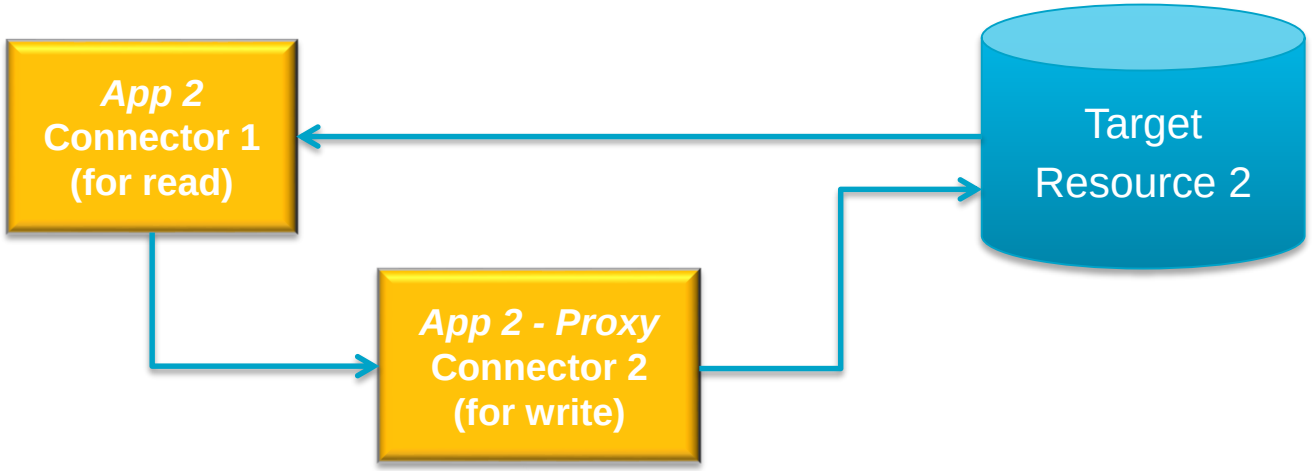
SailPoint

Provisioning Architecture

Connectors

- Direct

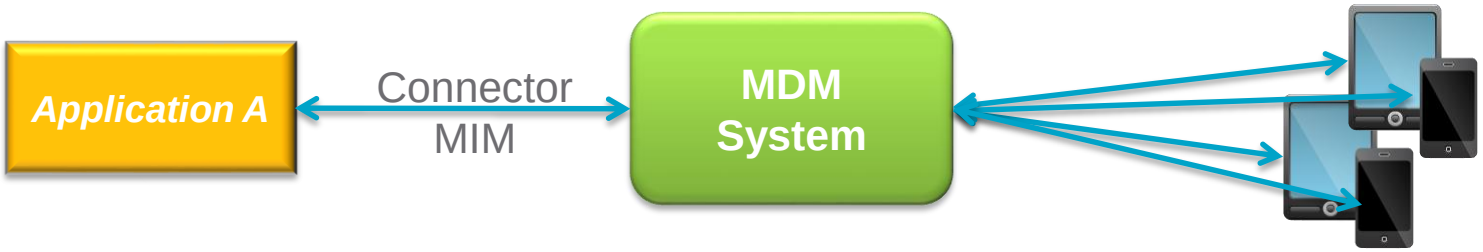
- Dual Channel (Proxy)



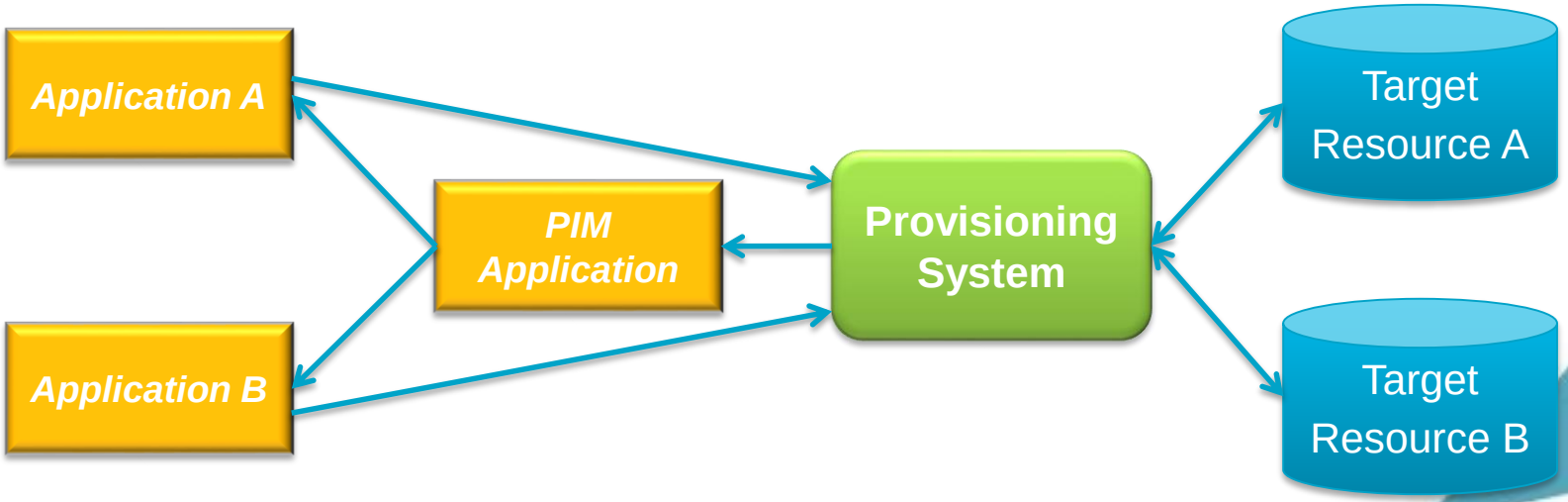
Provisioning Architecture

Integration Modules

- Direct – Mobile Device Management Integration Module (MIM)



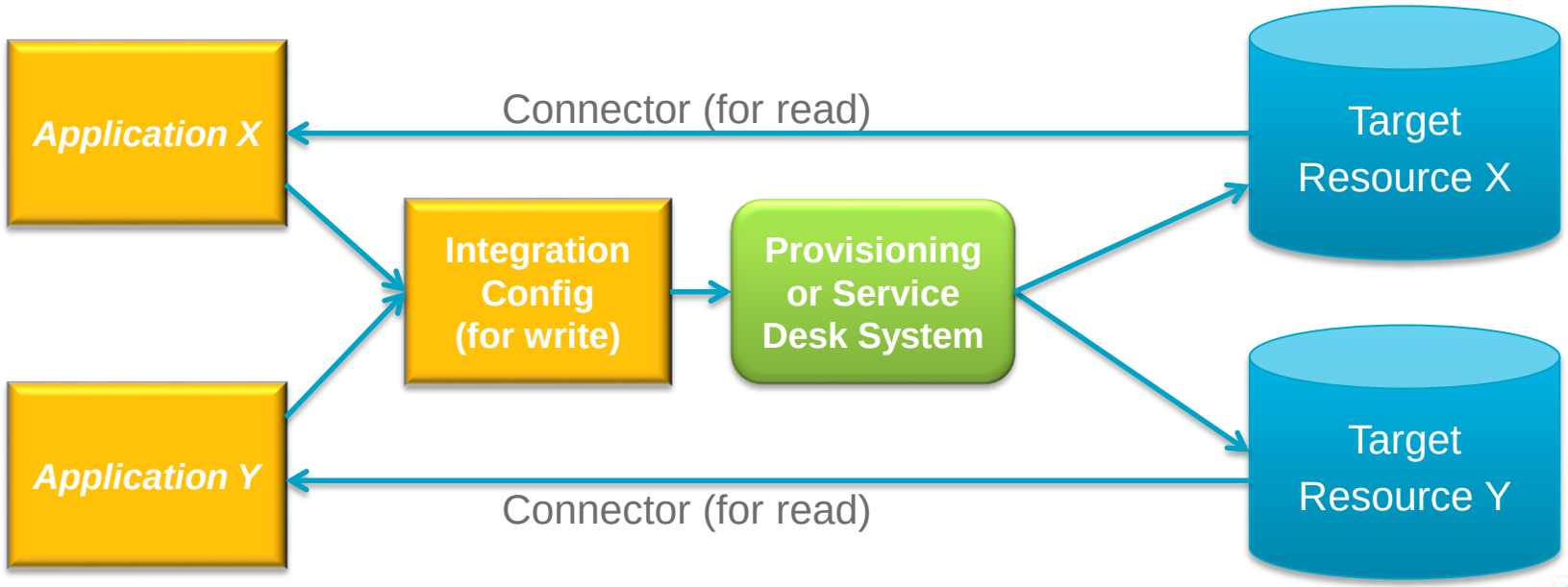
- Direct – Provisioning Integration Module (PIM)



Provisioning Architecture

Integration Modules

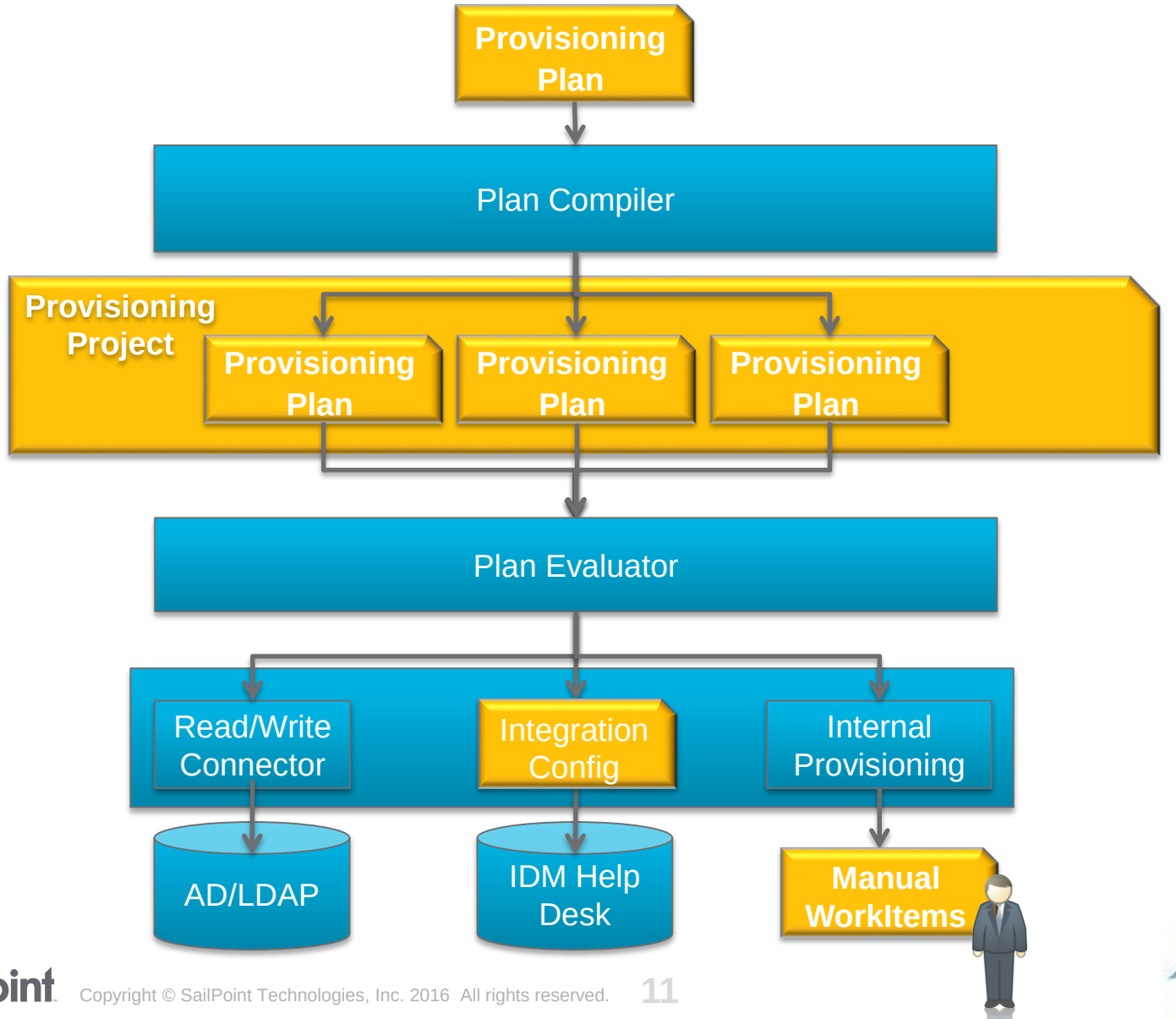
- Dual Channel (Integration Configuration)



Provisioning Process and Components

SailPoint

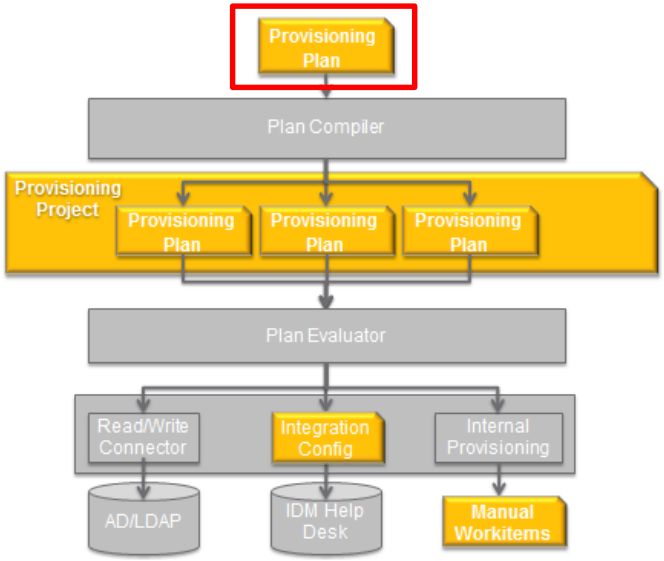
Provisioning Process



Provisioning Object

Provisioning Plan

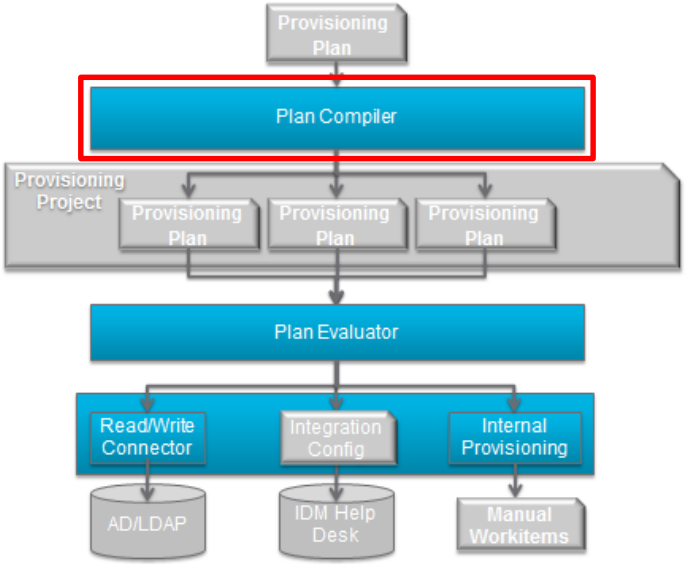
- Contains one or more requests for one identity
  - Role or entitlement request
  - Account request
  - Defines type of action
    - Create, modify, delete, unlock, enable, disable




```
<ProvisioningPlan>
  <AccountRequest application="LDAP"
nativeIdentity="cn=Andrea.Hudson,ou=people,dc=training,
dc=sailpoint,dc=com" op="Modify">
    <AttributeRequest name="groups" op="Add"
value="cn=VPN,ou=groups,dc=training,dc=sailpoint,dc=com"/>
  </AccountRequest>
...
```

Provisioning Component

Plan Compiler

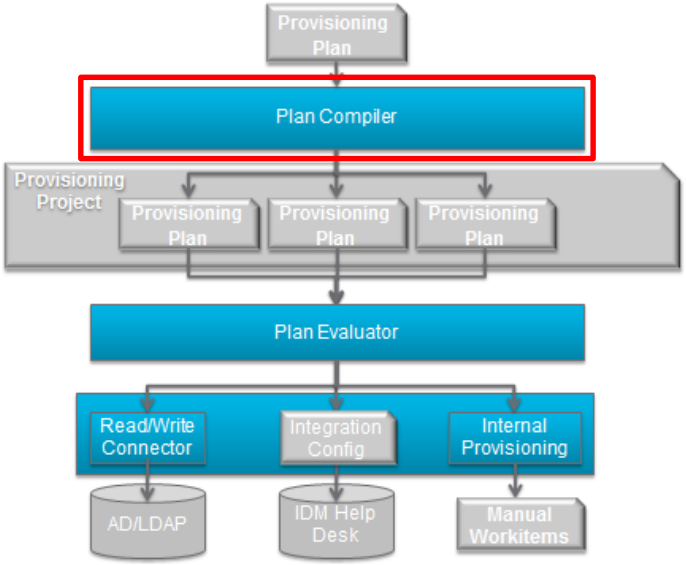


- Expands role requests into entitlement requests
- Compiles additional information for provisioning accounts or roles
  - Role Provisioning Policies
  - Application Provisioning Policies
  - Assimilates manual feedback
- Converts the original provisioning plan into a Provisioning Project

Provisioning Component

Plan Compiler (cont.)

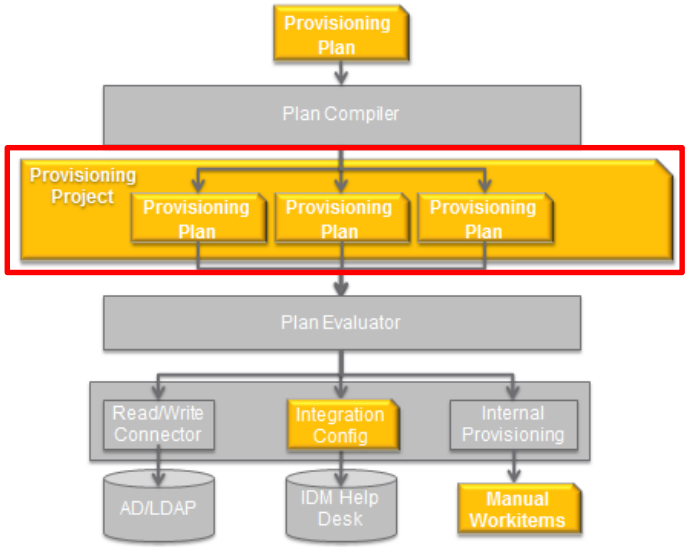
- Determines how provisioning will be carried out
  1. Integration Configuration with Managed Resources list
  2. Connector with provisioning configured
  3. Connector acting as proxy
  4. IdentityIQ work items



Provisioning Object

Provisioning Project

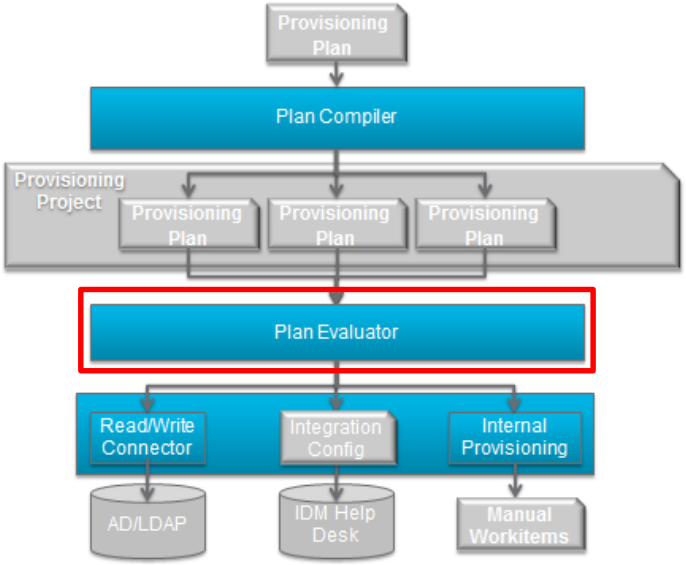
- Contains one or more provisioning plans split up by destination
- Contains copy of original plan



Provisioning Component

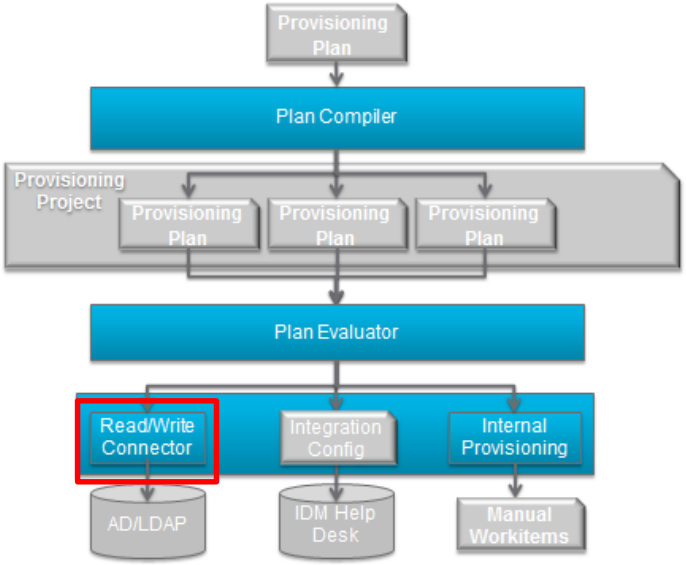
Plan Evaluator

- Input is a provisioning project object (or provisioning plan)
- Performs dependency checking
- Passes each plan to designated target
  - Connector, integration, or work item



Provisioning Component

Read/Write Connectors



- Used for aggregation and provisioning
- Implemented in IdentityIQ as Applications/Connectors
- Used by applications, PIMs, SIMs, and MIMs
- Can serve as provisioning proxy (application points to another application for provisioning)

Read/Write Connectors

Connector Registry

- Connector registry defines
  - Connector/integration support: featuresString
  - Class that handles provisioning
- Class defined as connector has a provision method that takes a single argument: a ProvisioningPlan
- Connector entry copied into Application object

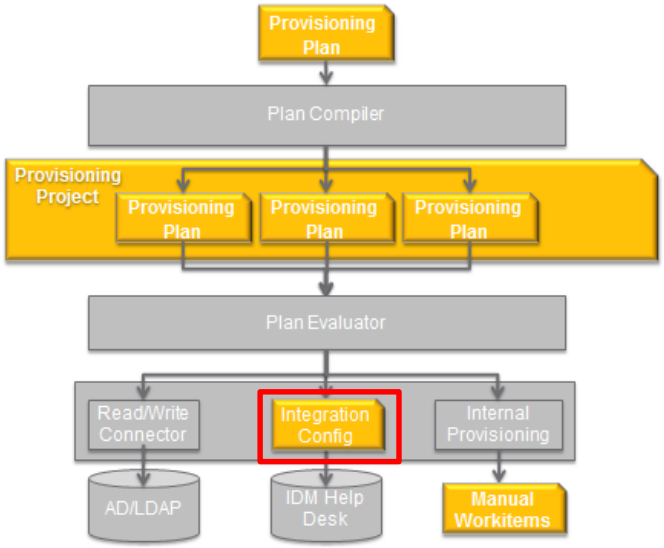
- Location
  - <install dir>/WEB-INF/config/connectorRegistry.xml
  - Debug Page → Configuration → Connector Registry

- Example

  <Application connector="sailpoint.connector.JDBCConnector" featuresString="DISCOVER_SCHEMA, **PROVISIONING**, GROUP_PROVISIONING" name="JDBC Template" type="JDBC">

Provisioning Object

Integration Configurations



- Defines parameters for how an integration runs
  - Details the integration executor class
  - Includes list of applications handled by IntegrationConfig
- Used by SIMs and some PIMs
- Installed through console or debug page
  - XML object loaded when enabling PIM/SIM support

IntegrationConfig Object

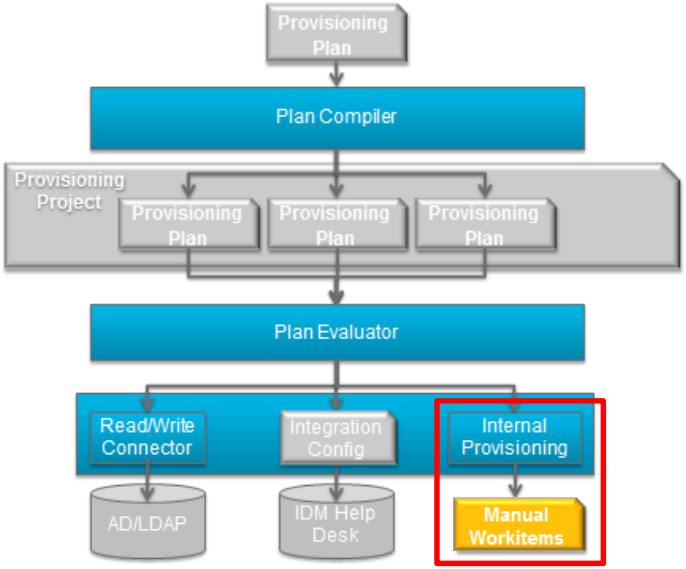
Example

```
<IntegrationConfig name="TDI Integration”
executor="sailpoint.integration.tdi.TDIIntegrationExecutor”>
…
<ManagedResources>
   <ManagedResource name="Active Directory">
     <ApplicationRef>
       <Reference class="sailpoint.object.Application" name="TDI Active Directory"/>
     </ApplicationRef>
   </ManagedResource>
</ManagedResources>
…
</IntegrationConfig>
```

Provisioning Component and Object

Internal Provisioning

- WorkItem delivered to user inbox
  - Provides details for manual provisioning
- Used when no provisioning pathway is defined



Universal Manager Setting

- Used to provision all non-specified applications
- Replaces work item as default channel
- Implemented through connector or integration configuration

- Provisioning Evaluation Process
    1. Integration Configuration with Managed Resources list
    2. Connector with provisioning configured
    3. Connector acting as proxy
    4. Integration Configuration with Universal Manager option enabled

Provisioning Process Walkthrough

SailPoint

Example: Plan (Original)

```
<ProvisioningPlan>
  <AccountRequest application="LDAP"
nativeIdentity="cn=Andrea.Hudson,ou=people,
dc=training,dc=sailpoint,dc=com" op="Modify">
    <AttributeRequest name="groups" op="Add"
value="cn=VPN,ou=groups,dc=training,dc=sailpoint,d
c=com"/>
  </AccountRequest>
  <Requesters>
    <Reference class="sailpoint.object.Identity"
id="2c901c0d31d3af460131d4854fa60424"
name="Randy.Knight"/>
  </Requesters>
</ProvisioningPlan>
```

Example: Do we have an Account?

- If not, we need to add Account Request
- Use Application Account Provisioning Policy to gather or populate fields

```
<AccountRequest application="LDAP"
nativeIdentity="cn=Andrea.Hudson,ou=people,dc=training,
dc=sailpoint,dc=com" op="Create">
  <AttributeRequest name="groups" op="Add" value="cn=VPN,
ou=groups,dc=training,dc=sailpoint,dc=com"/>
<AttributeRequest name="userPassword" op="Add" value="password">
      <Attributes>
        <Map>
          <entry key="secret" value="true"/>
        </Map>
      </Attributes>
    </AttributeRequest>
  <AttributeRequest name="cn" op="Add" value="Andrea.Hudson"/>
  <AttributeRequest name="ObjectClass" op="Add"
value="inetOrgPerson"/>
</AccountRequest>
```

Example: Provisioning Policy

- How do account fields get created?
  - Role and Application Provisioning Policy Defines how provisioning fields get populated
  - Can request users to fill in fields
    - Define owners
    - Auto-create forms
  - Can auto-populate fields
    Example:

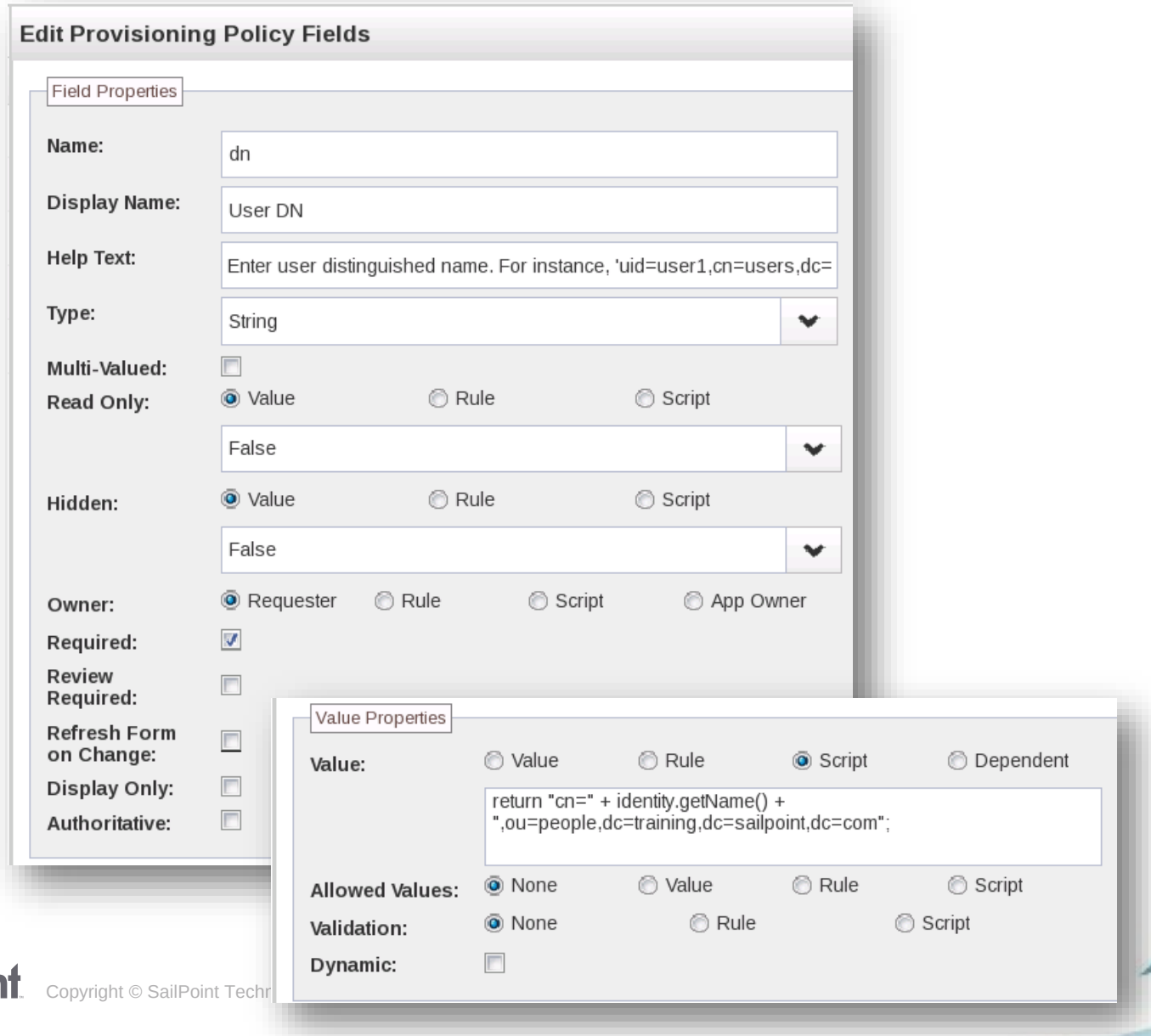
LDAP Provisioning Policy, Native Identity Attribute:

```
return "cn=" + identity.getName() + ",ou=people,
dc=training,dc=sailpoint,dc=com";
```

LDAP Provisioning Plan:

```
<AccountRequest application="LDAP"
nativeIdentity="cn=Andrea.Hudson,ou=people,dc=training,
dc=sailpoint,dc=com op="Create">
```

Example: Provisioning Policy – Providing Values



Example: Provisioning Plan with Account

```
<ProvisioningPlan>
  <AccountRequest application="LDAP"
nativeIdentity="cn=Andrea.Hudson,ou=people,dc=training,
dc=sailpoint,dc=com" op="Create">
    <AttributeRequest name="groups" op="Add" value="cn=VPN,
ou=groups,dc=training,dc=sailpoint,dc=com"/>
    <AttributeRequest name="userPassword" op="Add" value="password">
      <Attributes> ... </Attributes>
    </AttributeRequest>
    <AttributeRequest name="cn" op="Add" value="Andrea.Hudson"/>
    <AttributeRequest name="ObjectClass" op="Add" value="inetOrgPerson"/>
  </AccountRequest>
  <Attributes>
    <Map>
      <entry key="requester" value="Randy.Knight"/>
      <entry key="source" value="LCM"/>
    </Map>
  </Attributes>
  <Requesters>
    <Reference class="sailpoint.object.Identity"
id="2c901c0d31d3af460131d4854fa60424" name="Randy.Knight"/>
  </Requesters>
</ProvisioningPlan>
```

Debugging

SailPoint

Debugging/Troubleshooting

- Log4j.properties
  - log4j.logger.sailpoint.api.Provisioner
  - log4j.logger.sailpoint.provisioning.PlanCompiler
  - log4j.logger.sailpoint.provisioning.PlanEvaluator
  - log4j.logger.sailpoint.provisioning.IIQEvaluator
  - Logging for Connectors involved in provisioning (look at Connector Registry or Integration Config.)

- Auditing
  - Provision
  - Manual Provisioning
  - Provisioning Complete
  - Provisioning Failure

Questions?

SailPoint