# 🚀 SailPoint IdentityIQ Implementation Guide

## Accessio RACF Connector Deployment

**Document Version**: 1.0  
**Last Updated**: June 2025  
**Target Audience**: SailPoint Administrators, Implementation Teams

## 📋 Table of Contents

1. [Prerequisites & Environment Setup](#prerequisites--environment-setup)
2. [Pre-Implementation Checklist](#pre-implementation-checklist)
3. [SailPoint IIQ Configuration](#sailpoint-iiq-configuration)
4. [Connector Installation](#connector-installation)
5. [Application Configuration](#application-configuration)
6. [Workflow Deployment](#workflow-deployment)
7. [Provisioning Rules Setup](#provisioning-rules-setup)
8. [Task Configuration](#task-configuration)
9. [Testing & Validation](#testing--validation)
10. [Go-Live Procedures](#go-live-procedures)
11. [Post-Implementation](#post-implementation)
12. [Troubleshooting](#troubleshooting)

## 🔧 Prerequisites & Environment Setup

### **SailPoint IdentityIQ Requirements**

| Component | Minimum Version | Recommended Version | Notes |
| --- | --- | --- | --- |
| **SailPoint IIQ** | 8.2 | 8.4+ | Latest patch level recommended |
| **Java** | 11 | 17 | OpenJDK or Oracle JDK |
| **Application Server** | Tomcat 9.0 | Tomcat 10.0+ | Or equivalent (WebLogic, JBoss) |
| **Database** | PostgreSQL 12+ | PostgreSQL 14+ | Or Oracle 19c+, SQL Server 2019+ |
| **Memory** | 8GB RAM | 16GB+ RAM | For production environments |
| **CPU** | 4 cores | 8+ cores | For production environments |

### **Network & Connectivity Requirements**

# Required Network Access  
SailPoint IIQ Server → Garancy API Server  
- Protocol: HTTPS  
- Port: 443 (or custom SOAP port)  
- Authentication: Username/Password  
- Firewall: Bidirectional communication required  
  
# Optional (for monitoring)  
SailPoint IIQ Server → SMTP Server  
- Protocol: SMTP/SMTPS  
- Port: 25/587/465  
- Authentication: As per email server configuration

### **Service Accounts Required**

1. **Garancy Service Account**
   * Purpose: API authentication for RACF operations
   * Permissions: User/Role lifecycle management
   * Format: svc\_sailpoint\_racf
2. **SailPoint Service Account**
   * Purpose: Application server and database access
   * Permissions: Read/Write to IIQ database, file system access
   * Format: svc\_identityiq

## ✅ Pre-Implementation Checklist

### **Environment Preparation**

* **SailPoint IIQ Environment Ready**
  + IIQ application deployed and accessible
  + Database connectivity verified
  + Admin console accessible
  + System health checks passed
* **Garancy API Access**
  + API endpoint URL confirmed
  + Service account credentials obtained
  + Network connectivity tested
  + API documentation reviewed
* **Security Approvals**
  + Security assessment completed
  + Firewall rules approved and implemented
  + Certificate management plan in place
  + Encryption requirements validated
* **Backup & Recovery**
  + IIQ database backup scheduled
  + Configuration backup procedures defined
  + Rollback plan documented
  + Recovery testing completed

## 🏗️ SailPoint IIQ Configuration

### **Step 1: System Configuration Updates**

<!-- Add to SystemConfiguration -->  
<entry key="accessioRacfConnector.enabled" value="true"/>  
<entry key="accessioRacfConnector.maxRetries" value="3"/>  
<entry key="accessioRacfConnector.timeout" value="30000"/>  
<entry key="accessioRacfConnector.cacheSize" value="1000"/>  
<entry key="accessioRacfConnector.debugEnabled" value="false"/>

### **Step 2: Global Settings Configuration**

<!-- Add to Global Settings -->  
<GlobalSettings>  
 <entry key="workflow.approval.timeout" value="7"/>  
 <entry key="workflow.notification.enabled" value="true"/>  
 <entry key="provisioning.retry.attempts" value="3"/>  
 <entry key="aggregation.batch.size" value="500"/>  
 <entry key="recertification.reminder.days" value="7,3,1"/>  
</GlobalSettings>

### **Step 3: Email Configuration**

<!-- Email Template Configuration -->  
<EmailTemplate name="AccessioRACF-UserCreation">  
 <Subject>User Account Creation Request - $(identity.name)</Subject>  
 <Body>  
 <![CDATA[  
 Dear $(approver.displayName),  
   
 A new user account creation request requires your approval:  
   
 User: $(identity.displayName)  
 User ID: $(identity.name)  
 Organization: $(identity.getAttribute("organization"))  
 Request Type: $(workflowCase.type)  
 Requested By: $(launcher.displayName)  
   
 Please review and approve/reject this request in SailPoint IdentityIQ.  
   
 Best regards,  
 SailPoint IdentityIQ System  
 ]]>  
 </Body>  
</EmailTemplate>

## 📦 Connector Installation

### **Step 1: Deploy Connector JAR**

# 1. Stop SailPoint IIQ application server  
sudo systemctl stop tomcat  
  
# 2. Copy connector JAR to WEB-INF/lib  
cp accessio-racf-connector-1.0.0.jar $TOMCAT\_HOME/webapps/identityiq/WEB-INF/lib/  
  
# 3. Set proper permissions  
chown tomcat:tomcat $TOMCAT\_HOME/webapps/identityiq/WEB-INF/lib/accessio-racf-connector-1.0.0.jar  
chmod 644 $TOMCAT\_HOME/webapps/identityiq/WEB-INF/lib/accessio-racf-connector-1.0.0.jar  
  
# 4. Start SailPoint IIQ application server  
sudo systemctl start tomcat

### **Step 2: Verify Connector Installation**

# Check application logs for connector loading  
tail -f $TOMCAT\_HOME/logs/catalina.out | grep -i "accessio"  
  
# Expected output:  
# INFO: Loading connector: com.sailpoint.connector.accessio.racf.AccessioRACFConnector  
# INFO: Accessio RACF Connector initialized successfully

### **Step 3: Import Connector Configuration**

# Using IIQ Console  
iiq console  
> import /path/to/AccessioRACF.xml  
> commit  
> exit  
  
# Or using REST API  
curl -X POST "https://your-iiq-server/identityiq/rest/objects/Application" \  
 -H "Content-Type: application/xml" \  
 -H "Authorization: Bearer $TOKEN" \  
 -d @AccessioRACF.xml

## 🔧 Application Configuration

### **Step 1: Create Application in SailPoint IIQ**

1. **Navigate to**: Setup → Applications → New Application
2. **Select Connector**: Accessio RACF Connector
3. **Configure Connection**:

<!-- Application Configuration -->  
<Application name="Accessio RACF" type="AccessioRACF">  
 <Attributes>  
 <Map>  
 <!-- Garancy API Configuration -->  
 <entry key="garancyEndpoint" value="https://your-garancy-server/api/soap"/>  
 <entry key="garancyUsername" value="%%GARANCY\_USERNAME%%"/>  
 <entry key="garancyPassword" value="%%GARANCY\_PASSWORD%%"/>  
 <entry key="timeout" value="30000"/>  
 <entry key="retryAttempts" value="3"/>  
   
 <!-- Performance Configuration -->  
 <entry key="cacheEnabled" value="true"/>  
 <entry key="cacheSize" value="1000"/>  
 <entry key="cacheTTL" value="300"/>  
   
 <!-- Security Configuration -->  
 <entry key="sslEnabled" value="true"/>  
 <entry key="certificateValidation" value="true"/>  
 <entry key="encryptionEnabled" value="true"/>  
   
 <!-- Workflow Configuration -->  
 <entry key="userCreationWorkflow" value="AccessioRACFUserCreationWorkflow"/>  
 <entry key="roleAssignmentWorkflow" value="AccessioRACFRoleAssignmentWorkflow"/>  
 <entry key="userModificationWorkflow" value="AccessioRACFUserModificationWorkflow"/>  
 <entry key="userDeletionWorkflow" value="AccessioRACFUserDeletionWorkflow"/>  
   
 <!-- Approval Configuration -->  
 <entry key="approvalRequired" value="true"/>  
 <entry key="approvalTimeout" value="7"/>  
 <entry key="escalationEnabled" value="true"/>  
 <entry key="escalationTimeout" value="3"/>  
 </Map>  
 </Attributes>  
   
 <!-- Schema Configuration -->  
 <Schemas>  
 <Schema objectType="account" displayAttribute="BASEUS\_SAM\_ID">  
 <AttributeDefinition name="BASEUS\_SAM\_ID" type="string" required="true"/>  
 <AttributeDefinition name="BASEUS\_DISPLAY\_NAME" type="string"/>  
 <AttributeDefinition name="BASEUS\_EMAIL" type="string"/>  
 <AttributeDefinition name="BASEUS\_C\_C01\_001" type="string"/>  
 <AttributeDefinition name="BASEORG\_ID" type="string"/>  
 <AttributeDefinition name="BASEUS\_STATUS" type="string"/>  
 <AttributeDefinition name="BASEUS\_CREATED\_DATE" type="date"/>  
 <AttributeDefinition name="BASEUS\_MODIFIED\_DATE" type="date"/>  
 </Schema>  
   
 <Schema objectType="group" displayAttribute="BASEUSRC\_ROLE">  
 <AttributeDefinition name="BASEUSRC\_ROLE" type="string" required="true"/>  
 <AttributeDefinition name="TECHDSP\_NAME" type="string"/>  
 <AttributeDefinition name="ROLE\_TYPE" type="string"/>  
 <AttributeDefinition name="ROLE\_DESCRIPTION" type="string"/>  
 <AttributeDefinition name="ROLE\_OWNER" type="string"/>  
 <AttributeDefinition name="ROLE\_STATUS" type="string"/>  
 </Schema>  
 </Schemas>  
</Application>

### **Step 2: Test Connection**

# Test connection from IIQ Debug Console  
import sailpoint.api.SailPointContext;  
import sailpoint.object.Application;  
import sailpoint.connector.Connector;  
  
Application app = context.getObjectByName(Application.class, "Accessio RACF");  
Connector connector = sailpoint.connector.ConnectorFactory.getConnector(app, null);  
connector.testConfiguration();

## 🔄 Workflow Deployment

### **Step 1: Import Workflow Definitions**

# Import all workflow XML files  
iiq console  
> import /path/to/workflows/AccessioRACFUserCreationWorkflow.xml  
> import /path/to/workflows/AccessioRACFRoleAssignmentWorkflow.xml  
> import /path/to/workflows/AccessioRACFUserModificationWorkflow.xml  
> import /path/to/workflows/AccessioRACFUserDeletionWorkflow.xml  
> import /path/to/workflows/workflow-config.xml  
> commit  
> exit

### **Step 2: Configure Workflow Variables**

<!-- Workflow Configuration Variables -->  
<WorkflowVariables>  
 <Variable name="approvalTimeout" value="7"/>  
 <Variable name="escalationTimeout" value="3"/>  
 <Variable name="notificationEnabled" value="true"/>  
 <Variable name="auditEnabled" value="true"/>  
 <Variable name="debugEnabled" value="false"/>  
</WorkflowVariables>

### **Step 3: Approval Configuration**

<!-- Approval Process Configuration -->  
<ApprovalProcess name="AccessioRACF-UserCreation">  
 <ApprovalStep name="LineManagerApproval">  
 <Approver>$(identity.manager)</Approver>  
 <Timeout>7</Timeout>  
 <EscalationRule>$(identity.manager.manager)</EscalationRule>  
 </ApprovalStep>  
   
 <ApprovalStep name="OrgUnitOwnerApproval">  
 <Approver>$(orgUnit.owner)</Approver>  
 <Timeout>7</Timeout>  
 <EscalationRule>$(orgUnit.deputy)</EscalationRule>  
 </ApprovalStep>  
</ApprovalProcess>

## 📝 Provisioning Rules Setup

### **Step 1: Create Before Provisioning Rule**

<!-- BeforeProvisioningRule.xml -->  
<Rule name="AccessioRACF-BeforeProvisioning" type="BeforeProvisioning">  
 <Source>  
 <![CDATA[  
 import sailpoint.object.\*;  
 import sailpoint.api.SailPointContext;  
 import com.sailpoint.connector.accessio.racf.security.InputValidator;  
   
 // Validate input data  
 if (plan != null) {  
 for (AccountRequest request : plan.getAccountRequests()) {  
 if (request.getOperation() == ProvisioningPlan.Operation.Create) {  
 // Validate user ID  
 String userId = request.getStringAttribute("BASEUS\_SAM\_ID");  
 InputValidator.validateUserId(userId);  
   
 // Validate email  
 String email = request.getStringAttribute("BASEUS\_EMAIL");  
 InputValidator.validateEmail(email);  
   
 // Determine workflow based on user type  
 String userType = request.getStringAttribute("BASEUS\_C\_C01\_001");  
 if ("AARID".equals(userType)) {  
 request.setArgument("workflow", "AccessioRACFUserCreationWorkflow");  
 } else if ("STC".equals(userType) || "Technical".equals(userType)) {  
 request.setArgument("workflow", "AccessioRACFTechnicalUserWorkflow");  
 }  
 }  
 }  
 }  
   
 return plan;  
 ]]>  
 </Source>  
</Rule>

### **Step 2: Create After Provisioning Rule**

<!-- AfterProvisioningRule.xml -->  
<Rule name="AccessioRACF-AfterProvisioning" type="AfterProvisioning">  
 <Source>  
 <![CDATA[  
 import sailpoint.object.\*;  
 import sailpoint.api.SailPointContext;  
   
 // Log provisioning results  
 if (result != null && result.getStatus() == ProvisioningResult.STATUS\_SUCCESS) {  
 auditLog.info("Accessio RACF provisioning completed successfully for: " +   
 result.getIdentity().getName());  
   
 // Send notification  
 sendNotification(result.getIdentity(), "Account provisioned successfully");  
 } else if (result != null && result.getStatus() == ProvisioningResult.STATUS\_FAILED) {  
 auditLog.error("Accessio RACF provisioning failed for: " +   
 result.getIdentity().getName() + " - " + result.getErrors());  
 }  
   
 return result;  
 ]]>  
 </Source>  
</Rule>

## 📅 Task Configuration

### **Step 1: Aggregation Task Setup**

<!-- AccessioRACFAggregationTask.xml -->  
<TaskDefinition name="Accessio RACF Account Aggregation" type="Account Aggregation">  
 <Attributes>  
 <Map>  
 <entry key="applications">  
 <value>  
 <List>  
 <String>Accessio RACF</String>  
 </List>  
 </value>  
 </entry>  
 <entry key="checkDeleted" value="true"/>  
 <entry key="checkHistory" value="true"/>  
 <entry key="promoteManagedAttributes" value="true"/>  
 <entry key="refreshManagersBeforePromotion" value="true"/>  
 </Map>  
 </Attributes>  
   
 <!-- Schedule: Daily at 2 AM -->  
 <Schedule>  
 <CronExpression>0 0 2 \* \* ?</CronExpression>  
 </Schedule>  
</TaskDefinition>

### **Step 2: Recertification Task Setup**

<!-- AccessioRACFRecertificationTask.xml -->  
<TaskDefinition name="Accessio RACF Access Recertification" type="Access Recertification">  
 <Attributes>  
 <Map>  
 <entry key="applications">  
 <value>  
 <List>  
 <String>Accessio RACF</String>  
 </List>  
 </value>  
 </entry>  
 <entry key="certificationName" value="Accessio RACF Quarterly Review"/>  
 <entry key="schedule" value="QUARTERLY"/>  
 <entry key="autoRevoke" value="true"/>  
 <entry key="revokeGracePeriod" value="7"/>  
 </Map>  
 </Attributes>  
   
 <!-- Schedule: Quarterly on 1st of quarter -->  
 <Schedule>  
 <CronExpression>0 0 8 1 1,4,7,10 ?</CronExpression>  
 </Schedule>  
</TaskDefinition>

### **Step 3: Performance Monitoring Task**

<!-- PerformanceMonitoringTask.xml -->  
<TaskDefinition name="Accessio RACF Performance Monitor" type="Generic">  
 <Attributes>  
 <Map>  
 <entry key="class" value="com.sailpoint.connector.accessio.racf.tasks.PerformanceMonitorTask"/>  
 <entry key="thresholds">  
 <value>  
 <Map>  
 <entry key="responseTime" value="5000"/>  
 <entry key="errorRate" value="5"/>  
 <entry key="throughput" value="100"/>  
 </Map>  
 </value>  
 </entry>  
 </Map>  
 </Attributes>  
   
 <!-- Schedule: Every 15 minutes -->  
 <Schedule>  
 <CronExpression>0 \*/15 \* \* \* ?</CronExpression>  
 </Schedule>  
</TaskDefinition>

## 🧪 Testing & Validation

### **Phase 1: Unit Testing**

# Run connector unit tests  
cd /path/to/connector/project  
mvn clean test  
  
# Expected results:  
# Tests run: 45, Failures: 0, Errors: 0, Skipped: 0  
# [INFO] BUILD SUCCESS

### **Phase 2: Integration Testing**

# Test connection to Garancy API  
iiq console  
> import sailpoint.api.SailPointContext;  
> import sailpoint.object.Application;  
> Application app = context.getObjectByName(Application.class, "Accessio RACF");  
> app.testConnection();  
  
# Expected output: "Connection successful"

### **Phase 3: Functional Testing**

| Test Case | Description | Expected Result | Status |
| --- | --- | --- | --- |
| **TC001** | Test user creation workflow | User created with approval | ⏳ |
| **TC002** | Test role assignment | Role assigned after approval | ⏳ |
| **TC003** | Test user modification | User updated successfully | ⏳ |
| **TC004** | Test user deletion | User deleted with role cleanup | ⏳ |
| **TC005** | Test aggregation | Accounts/roles synchronized | ⏳ |
| **TC006** | Test recertification | Campaign created and executed | ⏳ |
| **TC007** | Test error handling | Graceful error handling | ⏳ |
| **TC008** | Test performance | Response time < 5 seconds | ⏳ |

### **Phase 4: Security Testing**

# Security validation checklist  
- [ ] Credential encryption verified  
- [ ] SSL/TLS communication confirmed  
- [ ] Input validation tested  
- [ ] Audit logging verified  
- [ ] Access controls validated  
- [ ] Error messages sanitized

## 🚀 Go-Live Procedures

### **Pre-Go-Live Checklist**

* **Environment Validation**
  + Production environment configured
  + Network connectivity verified
  + Security controls implemented
  + Monitoring systems active
* **Data Migration**
  + Existing user data analyzed
  + Migration scripts prepared
  + Data validation completed
  + Rollback procedures tested
* **User Training**
  + Administrator training completed
  + End-user documentation provided
  + Support procedures established
  + Escalation paths defined

### **Go-Live Steps**

# 1. Final backup  
iiq console  
> export objects Application "Accessio RACF" /backup/pre-golive/  
> commit  
> exit  
  
# 2. Enable production configuration  
# Update application.properties  
production.mode=true  
debug.enabled=false  
audit.level=INFO  
  
# 3. Start monitoring  
# Enable application monitoring  
# Start log monitoring  
# Activate alerting  
  
# 4. Execute initial aggregation  
iiq task "Accessio RACF Account Aggregation"  
  
# 5. Validate results  
# Check aggregation logs  
# Verify account counts  
# Confirm data accuracy

### **Post Go-Live Monitoring**

# Monitor for first 24 hours  
- Application performance metrics  
- Error rates and response times  
- User activity and feedback  
- System resource utilization  
- Network connectivity status

## 📊 Post-Implementation

### **Performance Optimization**

<!-- Performance Tuning Configuration -->  
<PerformanceSettings>  
 <entry key="connector.pool.size" value="10"/>  
 <entry key="connector.timeout" value="30000"/>  
 <entry key="cache.enabled" value="true"/>  
 <entry key="cache.size" value="5000"/>  
 <entry key="cache.ttl" value="300"/>  
 <entry key="batch.size" value="500"/>  
 <entry key="thread.pool.size" value="20"/>  
</PerformanceSettings>

### **Monitoring & Alerting Setup**

# Log monitoring patterns  
tail -f $IIQ\_HOME/logs/sailpoint.log | grep -E "(ERROR|WARN|AccessioRACF)"  
  
# Key metrics to monitor:  
- Response time: < 5 seconds average  
- Error rate: < 1% of requests  
- Throughput: > 100 operations/hour  
- Memory usage: < 80% of allocated  
- CPU usage: < 70% average

### **Maintenance Procedures**

| Task | Frequency | Description |
| --- | --- | --- |
| **Log Rotation** | Daily | Rotate and archive log files |
| **Cache Cleanup** | Weekly | Clear expired cache entries |
| **Performance Review** | Weekly | Review metrics and optimize |
| **Security Scan** | Monthly | Vulnerability assessment |
| **Backup Verification** | Monthly | Test backup and restore |
| **Documentation Update** | Quarterly | Update procedures and guides |

## 🔧 Troubleshooting

### **Common Issues & Solutions**

#### **Connection Issues**

# Issue: Cannot connect to Garancy API  
# Symptoms: Connection timeout errors in logs  
# Solution:  
1. Verify network connectivity: telnet garancy-server 443  
2. Check firewall rules and proxy settings  
3. Validate SSL certificates  
4. Test with curl: curl -v https://garancy-server/api/soap

#### **Authentication Failures**

# Issue: Authentication failed  
# Symptoms: 401 Unauthorized errors  
# Solution:  
1. Verify service account credentials  
2. Check password expiration  
3. Validate account permissions in Garancy  
4. Test credentials manually

#### **Performance Issues**

# Issue: Slow response times  
# Symptoms: Operations taking > 10 seconds  
# Solution:  
1. Check network latency: ping garancy-server  
2. Review connection pool settings  
3. Analyze database performance  
4. Monitor JVM memory usage

#### **Workflow Failures**

# Issue: Approval workflows not triggering  
# Symptoms: Requests stuck in pending state  
# Solution:  
1. Check workflow configuration  
2. Verify approver assignments  
3. Review email notification settings  
4. Check workflow execution logs

### **Log Analysis**

# Key log locations  
$IIQ\_HOME/logs/sailpoint.log # Main application log  
$IIQ\_HOME/logs/sailpoint-audit.log # Audit events  
$IIQ\_HOME/logs/sailpoint-workflow.log # Workflow execution  
$TOMCAT\_HOME/logs/catalina.out # Application server log  
  
# Useful log patterns  
grep "AccessioRACF" $IIQ\_HOME/logs/sailpoint.log  
grep "ERROR" $IIQ\_HOME/logs/sailpoint.log | grep "AccessioRACF"  
grep "workflow" $IIQ\_HOME/logs/sailpoint-workflow.log

### **Emergency Procedures**

# Emergency rollback procedure  
1. Stop application server  
2. Restore previous connector JAR  
3. Restore previous configuration  
4. Restart application server  
5. Verify system functionality  
6. Notify stakeholders  
  
# Emergency contact information  
- SailPoint Support: +1-800-SAILPOINT  
- Internal IT Support: ext. 5555  
- Garancy Support: contact@garancy.com  
- Security Team: security@company.com

## 📞 Support & Resources

### **Documentation References**

* [SailPoint IdentityIQ Documentation](https://documentation.sailpoint.com)
* [Connector Development Guide](https://developer.sailpoint.com)
* [Workflow Configuration Guide](https://community.sailpoint.com)
* [Security Best Practices](https://www.sailpoint.com/security)

### **Training Resources**

* SailPoint University: <https://university.sailpoint.com>
* Connector Development Training
* Workflow Design Best Practices
* Security Configuration Guidelines

### **Community Support**

* SailPoint Community: <https://community.sailpoint.com>
* Developer Forums
* Best Practices Sharing
* Technical Q&A

## ✅ Implementation Checklist

### **Pre-Implementation** ✅

* Environment prepared and validated
* Prerequisites met and documented
* Security approvals obtained
* Backup procedures established

### **Implementation** 🔄

* Connector installed and configured
* Application created and tested
* Workflows deployed and validated
* Rules and policies implemented

### **Testing** ⏳

* Unit testing completed
* Integration testing passed
* Functional testing validated
* Security testing approved

### **Go-Live** ⏳

* Production deployment completed
* Initial aggregation successful
* Monitoring systems active
* User training delivered

### **Post-Implementation** ⏳

* Performance optimized
* Monitoring configured
* Documentation updated
* Support procedures established

**Document Prepared By**: SailPoint Professional Services  
**Review Date**: June 2025  
**Next Review**: September 2025

For questions or support, contact: [sailpoint-support@company.com](mailto:sailpoint-support@company.com)