SSF Tools: Server-Specific Task Launcher User Guide

Document Revision History

Revision Date	Written/Edited By	Comments
April 2017	Paul Wheeler	First release with SSD v4
December 2017	Paul Wheeler	Minor update for SSD v5

Table of Contents

Introduction

In a multi-server environment, IdentityIQ provides the ability to run a task on any available server that is configured for task execution. However, in versions of IdentityIQ earlier than 7.2 the out-of-the-box product does not provide an easy way to run a given task on a specified server. Furthermore, a partitioned task can be run across all servers that are configured for task execution, but there is no easy way in any version of IdentityIQ to specify a group of servers that should run a given task without modifying the configuration. It would sometimes be useful to be able to run a non-partitioned task on a single specified server, or a partitioned task on a group of specified servers. The Server-Specific Task Launcher provides a solution for this.

The Server-Specific Task Launcher may be useful in the following cases:

- Troubleshooting issues with a task and ensuring logs are always written to the same location
- Running tasks that export data to a local filesystem where it is important that the file is exported to the same server
- Aggregating files where there is no shared storage and the file will only be available on one server
- Running partitioned aggregation and refresh tasks on all available servers out of normal working hours, making UI servers temporarily available as task servers

The Task Launcher is compatible with non-partitioned tasks and with partitioned tasks for account aggregation, identity refresh and role propagation.

Components

The Server-Specific Task Launcher includes these files:

File name	Description
ServerSpecificTaskLauncher.java	The main Task Executor Java class
TaskLauncherService.java	A service for launching single-server non-partitioned tasks
ServerSpecificTaskLauncher_TaskDefinition.xml	The TaskDefinition object for the task
ServerSpecificTaskLauncher_ServiceDefinition.xml	The ServiceDefinition object for the service

Installation

The Server-Specific Task Launcher is installed by default with the SSD. If you are not using the SSD, the following procedure can be used to install it manually:

1. Compile the two Java classes and copy the resulting class files to: WEB-INF/classes/sailpoint/services/standard/tasklauncher
2. Import the two XML files using one of the following methods:
   - Open iiq console and use the 'import' command:

import <filepath>/ ServerSpecificTaskLauncher_TaskDefinition.xml
import <filepath>/ ServerSpecificTaskLauncher_ServiceDefinition.xml

- In the UI use the “Import from File” functionality under Global Settings (IdentityIQ 7.0 and later) or System Setup (previous versions)

3. Restart the application servers.

How the Server-Specific Task Launcher Works

The Server-Specific Task Launcher uses two different methods of launching a task on a specific server or servers depending on whether the task to be launched has partitioning enabled and whether a single server or multiple servers are selected to run it.

Single-Server Non-Partitioned Tasks

For a single-server non-partitioned task, the task launcher uses the TaskLauncherService, which runs on every server. The launcher task creates a Custom object with a name in this format:

<server name>_launchtask_<Unique ID>

The Custom object has the task name as an attribute.

When a server runs the TaskLauncher service (by default every 30 seconds) it checks for a Custom object that has a name starting with a concatentation of the name of the server that the service is running on and “_launchtask_” and, if it finds one, launches the task whose name is stored in the object directly on that server (bypassing the task scheduler).

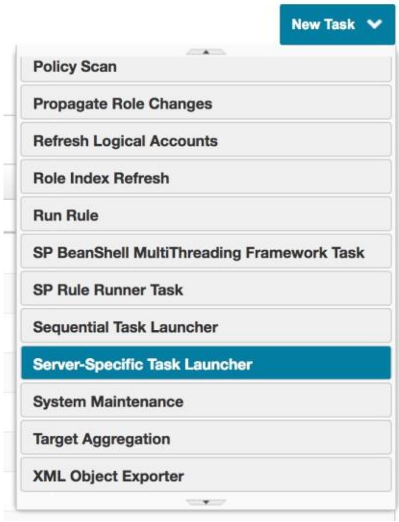
Multi-Server and/or Partitioned Tasks

If the task to be launched has partitioning enabled or if multiple servers are selected to run the task, the launcher task temporarily modifies the “Task” and “Request” ServiceDefinition objects. These each have a “hosts” property that defines which servers are permitted to run tasks or process partitions (amongst other things). The Server-Specific Task Launcher modifies the “hosts” property with the names of the servers specified by the user, but only until the task has launched and (where applicable) all partitions have been assigned hosts, after which the previous values are written back, the Task Launcher closes and the launched task continues running until completion on the specified server(s).

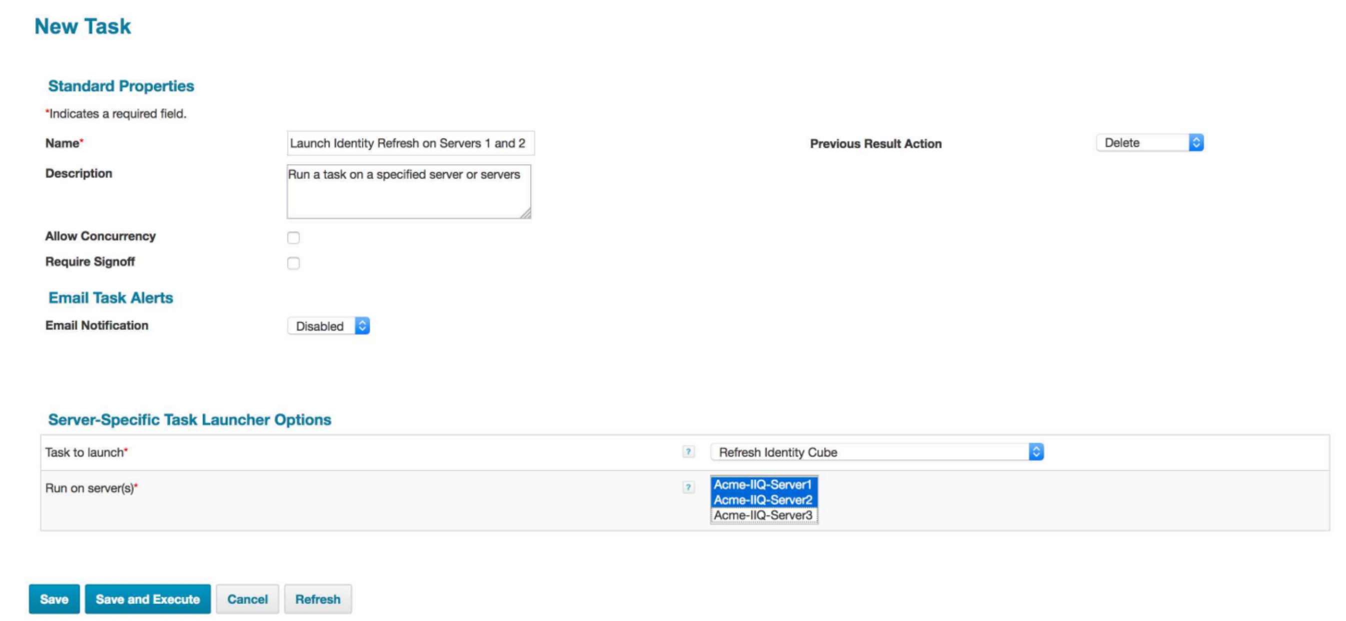
IdentityIQ only polls the ServiceDefinition objects once every 60 seconds, so when using this method of launching a task the Task Launcher must pause for 60 seconds after changing the values of the ServiceDefinition hosts before launching the task.

How to use the Server-Specific Task Launcher

In order to use the Task Launcher it is necessary to create a new instance of the launcher from the template Task Definitions. To do this, go to Setup -> Tasks (or Monitor -> Tasks in versions of IdentityIQ earlier than 7.0). Click “New Task” and create a new task instance for “Server-Specific Task Launcher”:



The New Task screen will appear for the Server-Specific Task Launcher as shown below:



New Task

Standard Properties

*Indicates a required field.

Name* Launch Identity Refresh on Servers 1 and 2

Previous Result Action Delete

Description Run a task on a specified server or servers

Allow Concurrency □

Require Signoff □

Email Task Alerts

Email Notification Disabled

Server-Specific Task Launcher Options

Task to launch*	Refresh Identity Cube
Run on server(s)*	Acme-IIQ-Server1 Acme-IIQ-Server2 Acme-IIQ-Server3

Give the Task Launcher instance a name, then in the “Task to launch” field select the task you want to launch from the dropdown. The “Run on servers” multi-select field lists all the servers that are currently online; select the server(s) from this list that you want to run the task on. For a non-partitioned task, selection of a single server will cause the task to be launched on that server, or if you select multiple servers any one of them will run it. If the selected task is partitioned, its execution may be shared by any or all of the selected servers. Save and execute the task, or save it and schedule for execution later.

The Task Results page for the Task Launcher will provide information on progress. When it has launched the selected task and has finished it will display the server(s) that the task was launched on.

Other Settings

The task has some hidden settings that are only editable in the Debug screen in IdentityIQ. The TaskDefinition for each Task Launcher instance can be edited to modify these settings.

Include Maintenance Partitions

By default, for partitioned aggregation tasks the Task Launcher will only ensure that the partitions that are concerned with the aggregation activities are launched on the selected servers since these are usually the partitions that are associated with most of the workload. When these partitions are completed an optional “Check Deleted Objects” partition and a “Finish Aggregation” partition are processed. However, it may take a long time for these maintenance partitions to start processing as they can only commence once the aggregation partitions have finished, and the hosts on which these maintenance partitions run is only determined after the aggregation partitions have completed. To avoid waiting for this and to ensure the Task Launcher can set the ServiceDefinition hosts back as early as possible and finish its work, the default setting is to not wait for the assignment of hosts on the maintenance partitions, so it is possible that they will run on servers other than those selected in the Task Launcher. Similarly, partitioned role propagation tasks are not monitored by default for their “Finish Role Propagation” partition to have a host assigned.

To force the Task Launcher to wait for assignment of hosts for maintenance partitions so that they will always run on the selected servers, add this to the attributes map in the TaskDefinition for the Task Launcher instance:

```
<entry key="includeMaintenancePartitions" value="true"/>
```

Task Start Timeout

If a multi-server or partitioned task does not launch promptly it may be a sign of an issue with that task, but the Task Launcher is designed to time out if the selected task does not launch within a reasonable time, by default 2 minutes. This helps to ensure the Task Launcher does not take too long to reset the original hosts values in the ServiceDefinition objects. To change this timeout, set the taskStartTimeout value in the TaskDefinition to the required number of seconds. For example, this will make it time out after 5 minutes

```
<entry key="taskStartTimeout" value="300"/>
```

Partitioned Task Finish Timeout

Some partitioned tasks can take a long time for all partitions to be assigned hosts, so the Task Launcher has a timeout after which it sets the original hosts values back in the ServiceDefinition objects, by default 2 hours. Any further assignment of hosts to partitions will use the original hosts values. To change this timeout, set the partitionedTaskFinishTimeout value in the TaskDefinition to the required number of seconds. For example, this will make it time out after 3 hours:

```
<entry key="partitionedTaskFinishTimeout " value="10800"/>
```

Limitations

The Server-Specific Task Launcher has the following limitations:

- For a single-server non-partitioned task, the task will launch when the server selected to launch the task runs the TaskLauncher service and queries for a Custom object intended for that server. There may be a short delay in this process because the service runs at a defined frequency (every 30 seconds by default). This is defined in the “interval” property of the TaskLauncher ServiceDefinition object. While the interval can be decreased by modifying this value, care should be taken when doing this. It is not recommended that this value be decreased to a value below 15 seconds, in order to avoid potential performance issues that might occur if it runs too frequently.
- For a multi-server or partitioned task, the Task Launcher must pause for 60 seconds before launching the selected task; this is to allow IdentityIQ time to poll for the new hosts settings in the ServiceDefinition objects.
- For a multi-server or partitioned task, when an instance of the Server-Specific Task Launcher is running it will be possible to run another instance to launch a different multi-server or partitioned task (either through setting the Task Launcher instance to allow concurrency or by creating a new Task Definition with a different name) but the second instance will wait until the first has completed before doing anything; this is because the first instance needs to set the original hosts values back in the ServiceDefinition object before the second instance changes them again to potentially different values.
- If any other task starts running (via a schedule or manually) while the Server-Specific Task Launcher is running for a multi-server or partitioned task, it will run on the same server(s) defined by the launcher because it will be referencing the same hosts values in the ServiceDefinition objects. If a task is started up to 60 seconds after the launcher has finished launching a multi-server or partitioned task it is possible that it will use the same servers defined by the launcher because of the 60 second frequency that IdentityIQ uses to poll the ServiceDefinition objects.
- The Server-Specific Task Launcher will not work for multi-server or partitioned tasks when the Task and Request servers are defined in the iiq.properties file using the environment.taskSchedulerHosts and environment.requestSchedulerHosts properties. This is

an older way of defining which servers are Task or Request servers, and this configuration should be moved into the ServiceDefinition objects for Task and Request.

- The Server-Specific Task Launcher will not work with versions of IdentityIQ earlier than 6.2.