How to Terminate Tasks

Scope

- The functions for terminating task processes for the JobScheduler Master and Universal Agent have been extended to allow the use of both SIGTERM and SIGKILL signals on Unix servers:
  - SIGTERM is sent first and allows an orderly termination of task processes to take place within a limited period of time.
  - If the time allowed has been exceeded and the processes are still running then SIGKILL will be sent.
- The information contained in this article draws together detailed information contained in a range of issues and should primarily be of interest to persons in engineering and to a lesser extent persons in operating functions.

Feature History

FEATURE AVAILABILITY STARTING FROM RELEASE 1.9

Related Issues

Support of this feature is subject to the following features:

- **JS-1495** - Getting issue details... STATUS
- **JS-1468** - Getting issue details... STATUS
- **JS-1463** - Getting issue details... STATUS
- **JS-1421** - Getting issue details... STATUS
- **JS-1420** - Getting issue details... STATUS
- **JS-1382** - Getting issue details... STATUS
- **JS-1307** - Getting issue details... STATUS
- **JS-1163** - Getting issue details... STATUS
- **JOC-10** - Getting issue details... STATUS

Use Case

Users who schedule programs and scripts that are aware of SIGTERM signals can implement clean-up procedures on receipt of the signal. Clean-up includes e.g. removal of temporary files, disconnect from a database and similar tasks.

- The use of both SIGTERM and SIGKILL signals on Unix servers has the following advantages:
  - The use of SIGTERM before SIGKILL means that there is a greater chance of data being saved after the signal has been issued.
  - The SIGTERM signal can - in contrast with SIGKILL - be monitored, i.e. a post-processing Script can be carried out.
  - The implementation of SIGTERM allows post-processing methods such as `spooler_process_after()`.
- The time allowed between the SIGTERM and SIGKILL signals can be specified in the command using the *timeout* attribute (the default is 15 sec): `<kill_task … timeout=".."/>`
- This feature can also be applied for:
  - remote processes, i.e. processes started by SSH and those started by an Agent,

- child processes started by a process running on an agent (**JS-1468** - Getting issue details... STATUS).

Implementation

JobScheduler Master

The following operations can be carried out from the JobScheduler Operating Center interface (JOC) or by use of the command line:

1. Operation: *kill immediately*
   - JOC sends `<kill_task immediately="yes"/>`
   - The process is killed immediately using the SIGKILL signal.
2. Operation: *terminate with timeout*
   - JOC sends `<kill_task immediately="yes" timeout="15"/>`
   - The process receives a SIGTERM signal. Should that process not terminate within the specified timeout period then it will be killed with a SIGKILL signal.
3. Operation: *terminate*
   - JOC sends `<kill_task immediately="yes" timeout="never"/>`
   - The process receives a SIGTERM signal. Monitoring of the process termination as described in Operation 2 above is not carried out.

See also

- **JOC-10** - Getting issue details... STATUS

JobScheduler Universal Agent (JUA)

The JobScheduler Universal Agent uses a script to monitor and kill parent and child tasks.

The script is to be found the `./bin` folder of the JUA installation.

The -kill-script option can be used to specify a different kill script if necessary. See the Command Line Options section in JobScheduler Universal Agent - Installation & Operation for more information.

See also

- **JS-1468** - Getting issue details... STATUS
- **JS-1495** - Getting issue details... STATUS

for more detailed information about the implementation.

Delimitation

- This feature is intended for Unix platforms that implement the SIGTERM and SIGKILL signals. It is not intended for Windows platforms for which exclusively the kill Immediately command applies.
- When using traps then please consider that the process created by the `<script ... language="shell"/>` element receives the signal. Subsequent scripts that are called within the `<script ... language="shell"/>` element will not receive the signal. You could therefore:
  - configure traps directly within the `<script ... language="shell"/>` element. The shell process will then receive and handle the signal.
  - configure traps in a shell script that is added by an `<include>` element instead of being stated within the `<script ... language="shell"/>` element. The included shell script will receive and handle the signal.
  - forward signals to subsequent shell scripts that are called within a `<script ... language="shell"/>` element.

- This feature has been fully implemented for the Universal Agent and for the Classic JobScheduler Agents (**JS-1420** - Getting issue details... STATUS).

Implementation Summary

The implementation of the different termination operations available for the JobScheduler Master and Universal Agent is summarized in the table below.

The links lead to the issues relevant to the implementation of the feature.

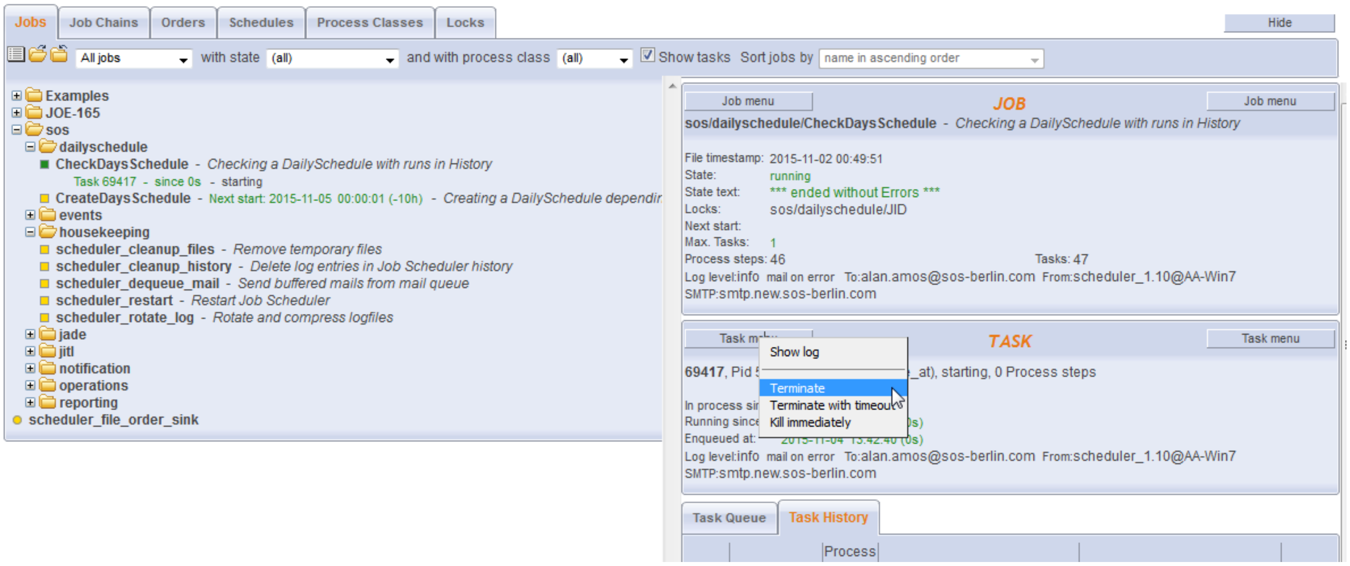
The termination operations available are:

- Terminate: `<kill_task immediately="true" timeout="...">`
- Kill: `<kill_task immediately="true">`
- Timeout: a job with the timeout attribute.

	Windows Standalone	Linux Standalone	Windows Classic Agent	Linux Classic Agent	Windows Universal Agent	Linux Universal Agent
Shell job						
Terminate	not supported	✓	not supported	✓ JS-1420	not supported	✓
Kill	✓	✓	✓	✓ JS-1421	✓ JS-1468	✓ JS-1468
Timeout	✓ JS-1463	✓ JS-1463	✓	✓		✓ JS-1468
Shell job with monitor						
Terminate	not supported	✓	not supported	✓	not supported	✓ JS-1468
Kill	✓	✓	✓	✓	✓ JS-1382	✓ JS-1468
Timeout	✓	✓				✓ JS-1382
API job						
Terminate	not supported	✓	not supported	✓	not supported	✓
Kill	✓	✓	✓	✓	✓	✓
Timeout	✓	✓				

Task Termination in JOC

The following screenshot shows how a task can be terminated using the Task menu in JOC's Task pane. This pane is shown in the Jobs tab when the task is running and the task can be terminated either with timeout or immediately.



All jobs with state (all) and with process class (all) ☑ Show tasks Sort jobs by name in ascending order

```
⊞ Examples
⊞ JOE-165
⊟ sos
  ⊟ dailyschedule
    CheckDaysSchedule - Checking a DailySchedule with runs in History
      Task 69417 - since 0s - starting
    CreateDaysSchedule - Next start: 2015-11-05 00:00:01 (-10h) - Creating a DailySchedule dependir
  ⊞ events
  ⊟ housekeeping
    scheduler_cleanup_files - Remove temporary files
    scheduler_cleanup_history - Delete log entries in Job Scheduler history
    scheduler_dequeue_mail - Send buffered mails from mail queue
    scheduler_restart - Restart Job Scheduler
    scheduler_rotate_log - Rotate and compress logfiles
  ⊞ jade
  ⊞ jitl
  ⊞ notification
  ⊞ operations
  ⊞ reporting
● scheduler_file_order_sink
```

```
sos/dailyschedule/CheckDaysSchedule - Checking a DailySchedule with runs in History

File timestamp: 2015-11-02 00:49:51
State:          running
State text:     *** ended without Errors ***
Locks:          sos/dailyschedule/JID
Next start:
Max. Tasks:     1
Process steps: 46                                   Tasks: 47
Log level:info  mail on error  To:alan.amos@sos-berlin.com  From:scheduler_1.10@AA-Win7
SMTP:smtp.new.sos-berlin.com
```

```
69417, Pid 5          _at), starting, 0 Process steps

In process si
Running since                 0s)
Enqueued at:    2015-11-04 13:42:40 (0s)
Log level:info  mail on error  To:alan.amos@sos-berlin.com  From:scheduler_1.10@AA-Win7
SMTP:smtp.new.sos-berlin.com
```

Examples

Using a SIGTERM trap to show the difference between the `kill` and `terminate` commands

Download the Example

job_trap_sigterm.job.xml.zip

Description

This example contains a job that uses a SIGTERM trap to show the difference between the `<kill_task>` and `<terminate_task>` commands provided by JOC.

The job `job_trap_sigterm.job.xml` shows how to trap the terminate command provided by JOC.

- Start the job
- Terminate the task in JOC
- You will see the log message: `sigterm will be ignored`
- The task will continue

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<job  title="test test">
    <script  language="shell">
        <![CDATA[
trap 'echo sigterm will be ignored' 15
for i in 1 2 3 4 5 6 7 8 9 0
do
date
sleep 10
done
sleep 60
        ]]>
    </script>

    <run_time />
</job>
```

Change Management References

T	Key	Linked Issues	Fix Version/s	Status	P	Summary	Updated
⊞	JS-1552	JS-1550 , JS-1551 , JS-1558	1.10.3, 1.11	RELEASED	⇊	Agent start script executes a shell script to kill running tasks in case of a JobScheduler Agent crash	Jan 15, 2016
⊞	JS-1551	JS-1550 , JS-1572 , JS-1552 , JS-1574 , JS-1468	1.10.3, 1.11	RELEASED	⇊	JobScheduler Agent maintains a list of running tasks	Jan 16, 2016
⊞	JS-1550	JS-1551 , JS-1552 , JS-1549	1.10.3, 1.11	RELEASED	⇊	Tasks should terminate in case of JobScheduler Agent crash	Jan 15, 2016
⊞	JS-1549	JS-1550	1.10.3, 1.11	DISMISSED	⇊	Tasks should terminate in case of JobScheduler Master crash	Jan 27, 2016
⊞	JS-1495	DOC-35 , JS-1468	1.10	RELEASED	⇊	Integrate task kill features into JobScheduler Universal Agent start script	Nov 18, 2015
⊞	JS-1468	JS-1382 , JS-1495 , JS-1551 , DOC-35 , JS-1558	1.10	RELEASED	⇊	OS-specific kill for JobScheduler Universal Agent Tasks without monitor	Dec 16, 2015
●	JS-1463	DOC-35	1.9.5, 1.10	RELEASED	⇊	job timeout on a JobScheduler Master kills a shell task after the specified timeout	Nov 30, 2015
●	JS-1421	DOC-35	1.9.4, 1.10	RELEASED	⇈	The <kill_task immediately="yes"> command on a shell task without monitor on classic Agent kills all children of the shell task	Nov 30, 2015
⊞	JS-1420	JS-1163 , DOC-35	1.10	RELEASED	⇊	SIGTERM on shell task on classic agent over tcp	Nov 18, 2015
⧉	JS-1382	JS-1386 , DOC-35 , JS-1468	1.10	DISMISSED	⇊	JobScheduler Universal Agent: kill a running task (including child processes)	Nov 30, 2015
⊞	JS-1307	JS-1163 , JS-1496 , DOC-35	1.9	RELEASED	⇈	SIGTERM on shell task with monitor is forwarded to shell process	Nov 30, 2015
⊞	JS-1163	JOC-10 , JS-1420 , DOC-35 , JOC-69 , JS-1307	1.9	RELEASED	⇈	<kill_task timeout="..."> with SIGTERM before SIGKILL	Mar 01, 2016

⊞	JOC-10	JS-1163 , DOC-35	1.9	RELEASED	⏫	A new command to terminate a task with SIGTERM should exist	Nov 20, 2015

13 issues

References

- JobScheduler Reference Documentation:
    - the <kill_task> XML command