Cross-Platform Scheduling

- Introduction
- Agents
- Agentless Scheduling
- Resources
  - Feature in Detail
  - How To ... Articles
  - Frequently Asked Questions
  - Examples in Detail

Introduction

JobScheduler uses two methods for remote execution: JobScheduler Agents and Agentless Scheduling per SSH.

- Agents are used in a Master / Agent Cluster and are completely controlled by a Master JobScheduler. They are used for simplified roll-out do and not require an individual configuration and database connection.
- Remote execution by SSH does not require a JobScheduler component on the remote server - instead an existing SSH server is used to create a secure shell and to execute commands and programs.
- See our What is the difference between SSH job execution and remote JobScheduler Agents? article for more information.

Agents

Any number of JobScheduler Universal Agents are controlled by a Master JobScheduler in a Master / Agent Cluster and are used for carrying out remote scheduling tasks.

- Agents are equipped with a reduced set of functions that allows simplified roll-out and configuration.
- Agents do not require a database connection and do not dispose of individual configuration files as job execution on an Agent is triggered by its Master and the Agent logs the execution result to its Master.

Agentless Scheduling

The SSH (Secure Shell) JITL job (JobSchedulerSSHJob) allows a JobScheduler to execute programs on another computer without a JobScheduler Agent being installed on that machine.

The JobSchedulerSSHJob allows execution on Windows and Unix systems.

- The elegance of SSH is its simplicity. It allows public/private key authentication and is well suited to executing programs for specific accounts.
  - The JobScheduler SSH JITL job allows shell scripts to return values to JobScheduler orders as parameters.
  - See How To - Usage of the SSH Job (JobSchedulerSSHJob) with JCraft's JSch.
- One of the restrictions of standard SSH connections is that there is limited control of child processes on the remote host.
  - If a number of child processes are spawned by a program during an SSH session and that session gets killed then these child processes would normally continue running.
  - The JobScheduler provides a method for monitoring SSH connections that allows both remote sessions and local JobScheduler tasks to be terminated.
  - See How To - SSH Session Management for more information.
- The JobScheduler provides a means of controlling the behavior of standard error output and exit codes of shell commands using the SSH JITL Job.
  - See Job JobSchedulerSSHJob - Handling StdErr and ExitCode for more information.

Read more ....

Resources

Feature in Detail

- How to dynamically specify a remote JobScheduler instance

How To ... Articles

- How to carry out OS-independent file operations
- How to configure remote execution
- How to dynamically assign JobScheduler Agents to a job
- How to dynamically specify a remote JobScheduler instance
• How to get the PID of a task executed by a remote JobScheduler Agent
• How to implement Central Configuration with a Supervisor JobScheduler
• How to organise secure file transfer when files become available on a remote host

Frequently Asked Questions
• Can I edit remote JobScheduler objects per UNC?
• How to execute Jobs and Orders with Remote JobScheduler Instances?
• What is the difference between SSH job execution and remote JobScheduler Agents?
• Which components will be installed on a client and how will jobs be started there?

Examples in Detail
• Job JobSchedulerSSHJob - Handling StdErr and ExitCode

Pages
• Agents
  • Master / Agent Compatibility
• Agentless Scheduling
  • Agentless scheduling on Windows with copSSH
  • Agentless scheduling on Windows with PsExec
  • Agentless scheduling with the JobScheduler SSH Job
  • Limitations when using SSH jobs for Windows

Navigation