

Gestion de tâches centralisées avec RunDeck



RunDeck est un outil centralisé de gestion de tâches (cron par exemple). Il est écrit en Java et gère la connexion vers ses «nodes» en SSH. C'est un outil très puissant avec divers plugins intéressants permettant entre autre l'interconnexion avec des outils de gestion de configurations (Salt, Puppet ...).

Nous allons donc voir comment mettre en place un serveur RunDeck et le relier à différents nodes afin de programmer divers tâches. Dans la plupart des cas, il est utilisé pour automatiser des déploiements ou des tests unitaires cependant on peut également l'utiliser pour des tâches classiques utilisées dans les cron jobs.

On va utiliser le paquet *.deb fournit par le site officiel pour une installation sur un serveur Debian / Ubuntu :

```
wget dl.bintray.com/rundeck/rundeck-deb/rundeck-2.6.2-1-GA.deb
```

```
apt-get install openjdk-7-jdk
```

```
dpkg -i rundeck-2.6.2-1-GA.deb
```

L'installation est terminée !

RunDeck fournit une interface Web, une interface en ligne de commande et une API Web. Nous allons utiliser principalement l'interface web pour gérer nos tâches.

Le service web écoute par défaut sur **localhost**, on va donc éditer le fichier de configuration **/etc/rundeck/rundeck-config.properties** et modifier cette ligne :

```
# change hostname here  
grails.serverURL=http://your-server:4440
```

On va maintenant redémarrer RunDeck pour qu'il prennent cela en compte :

```
service rundeckd restart
```

Il faut attendre un petit peu que le service soit démarré et vous devriez pouvoir joindre l'interface web sur le port 4440. Le login par défaut est **admin;admin**, n'oubliez pas de le changer.

On va donc commencer par créer un projet :

RunDECK

Create a new Project

Project Name
synchronisation

Description
synchronisation (top.Racktables.OM.netdisco)

Execution Mode
Job execution and schedule configuration at project level.

☐ Disable Execution
☐ Disable Schedule

User Interface
Additional configuration for the user interface for this project

Job Group Expansion Level
1
In the Jobs page, expand Job groups to this depth by default. More >

Display the Project Readme
☐ Projects List
☐ Project Home Page

Display the Project MOTD
☐ Projects List
☐ Project Home Page

Laissez les paramètres par défaut si ils vous conviennent, ce sera très bien.

RunDeck utilise une clé privée SSH pour se connecter à ses nodes, elle est situé à **/var/lib/rundeck/.ssh/id_rsa**.

La liste des nodes se situe dans le fichier

/var/rundeck/projects/synchronisation/etc/resources.xml, c'est ici qu'on ajoute les serveurs qu'on souhaitent administrer.

```
<?xml version="1.0" encoding="UTF-8"?>

<project>
  <node name="rundeck-win-pp" description="Rundeck server node" tags="" hostname="rundeck-
win-pp.in.ac-versailles.fr" osArch="amd64" osFamily="unix" osName="L$

  <node name="racktables-server" description="Racktables node" tags="" hostname="racktables-
server.in.ac-versailles.fr" osArch="amd64" osFamily="unix" osNam$

</project>
```

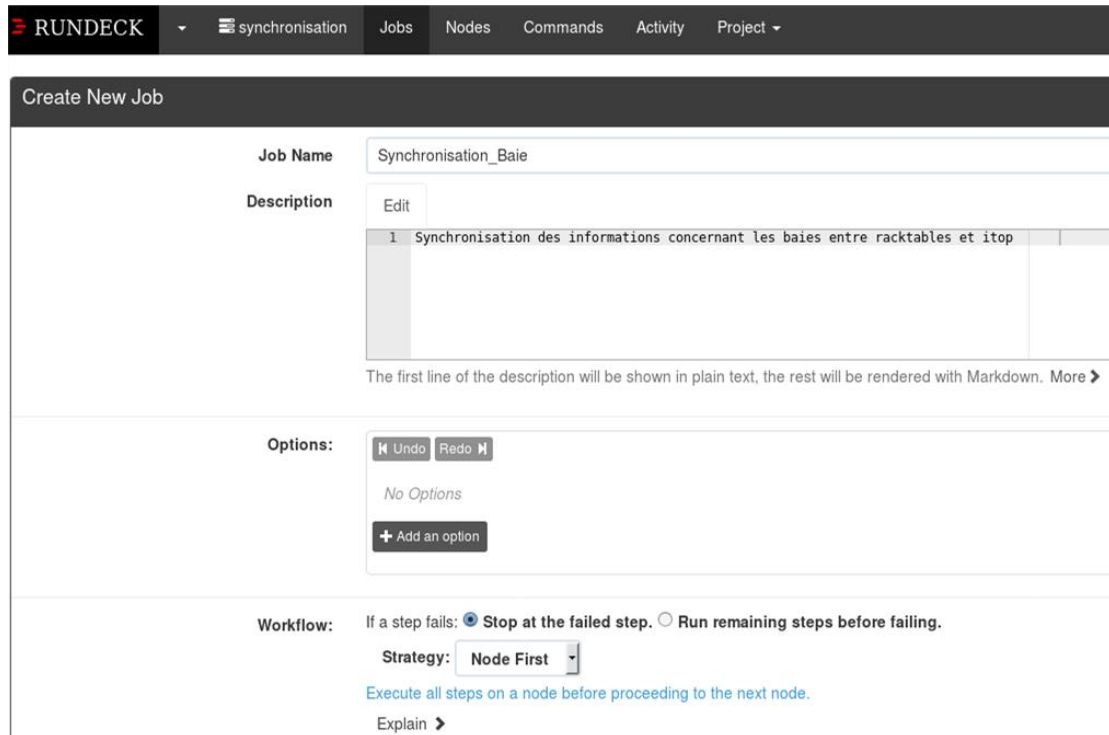
Dans cet exemple de projet, le serveur RunDeck est relié au serveur racktables et à lui même (pour gérer ses propres tâches). Il suffit maintenant d'ajouter la clé publique du serveur RunDeck sur le serveur racktables à manager:

```
ssh-copy-id -i /var/lib/rundeck/.ssh/id_rsa.pub root@racktables-server
```

Le serveur est donc joignable par Rundeck sans saisir de mot de passe ! On peut vérifier avec cette commande qui devrait vous connecter directement :

```
ssh root@racktables-server -i /var/lib/rundeck/.ssh/id_rsa
```

Maintenant qu'ils sont reliés, on va pouvoir créer nos premières tâches à effectuer sur ces serveurs :



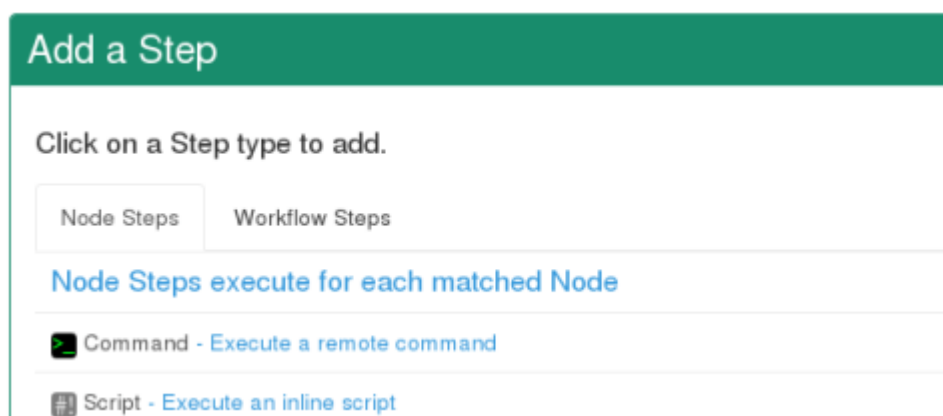
The screenshot shows the 'Create New Job' interface in Rundeck. The top navigation bar includes 'RUNDECK', 'synchronisation', 'Jobs', 'Nodes', 'Commands', 'Activity', and 'Project'. The form has the following sections:

- Job Name:** A text input field containing 'Synchronisation_Baie'.
- Description:** A section with an 'Edit' button and a table containing one step:

Step	Description
1	Synchronisation des informations concernant les baies entre racktables et itop

Below the table, a note states: 'The first line of the description will be shown in plain text, the rest will be rendered with Markdown. More >'
- Options:** A section with 'Undo' and 'Redo' buttons, the text 'No Options', and an 'Add an option' button.
- Workflow:** A section with a radio button selection for 'If a step fails:' (selected: 'Stop at the failed step', unselected: 'Run remaining steps before failing'), a 'Strategy:' dropdown menu set to 'Node First', and a link 'Execute all steps on a node before proceeding to the next node.' with an 'Explain >' link below it.

Le fonctionnement des Jobs sous RunDeck s'effectue en étapes (Steps), elles peuvent être l'exécution d'une commande, la lancement d'un script à distance ou même l'exécution d'un autre job. On ajoute donc une étape pour exécuter un script à distance :



The screenshot shows the 'Add a Step' dialog box. It has a green header with the title 'Add a Step'. Below the header, it says 'Click on a Step type to add.' There are two tabs: 'Node Steps' (selected) and 'Workflow Steps'. Under the 'Node Steps' tab, there are two options:

- Command - Execute a remote command** (indicated by a terminal icon)
- Script - Execute an inline script** (indicated by a document icon)

Nous allons prendre l'exemple que nous souhaitons lancer la commande `./baie.sh` qui permet d'exécuter le script de synchronisation des données concernant les emplacements des serveurs dans le baie. Pour cela on ajoute donc une étape «**Command**».

Global Log Filters:

[+ add](#)

1.

Command	<code>./home/administrateur/script_sync_baie/baie.sh</code>
Step Label	script baie

No Workflow steps

On a ensuite la possibilité:

-de choisir sur quel node l'exécuter (**locally** ou **dispatch to nodes**), on va choisir d'exécuter le script sur le nœud **racktables-server**

Nodes ☒ Dispatch to Nodes ☐ Execute locally

Choose whether the Job will run on filtered nodes or only on the local node.

Node Filter ▼ racktables-server

Matched Nodes

1 Node Matched

 racktables-server

-de choisir une fréquence de répétition (**schedule to run**), on va programmer l'exécuter du script quotidiennement à 5h30. On peut bien sur utiliser une syntaxe Crontab à la place.

Schedule to run repeatedly? ☐ No ☒ Yes

Simple ☐ Crontab

05 : 30

☒ Every Day ☒ Every Month

-de recevoir des notifications ou de relancer en cas d'échec, on va configurer le rundeck pour qu'on reçoit une notification lors d'un échec.

Send Notification? ☐ No ☒ Yes

On Success

☐ Send Email

☐ Webhook

☐ Slack Incoming WebHook Sends Rundeck Notifications to Slack

On Failure

☒ Send Email

To:

Subject:

☐ Webhook

☐ Slack Incoming WebHook Sends Rundeck Notifications to Slack

On Start

☐ Send Email

☐ Webhook

☐ Slack Incoming WebHook Sends Rundeck Notifications to Slack

Average Duration Exceeded

☐ Send Email

☐ Webhook

☐ Slack Incoming WebHook Sends Rundeck Notifications to Slack

Maintenant que notre job est créé, on va pouvoir le trouver dans notre liste de jobs et l'exécuter :

Synchronisation Baie

Action in 16h2m

Synchronisation des informations concernant les baies entre racktables et itop

Prepare and Run...

Definition

Input Options

None for this Job.

Log level

☒ Normal ☐ Debug

Debug level produces more output

Nodes

☐ Change the Target Nodes (!)

Run Job Now

Run Job Later

☒ Follow execution

Statistics

EXECUTIONS

0

Activity for this Job

☒ running ☐ recent ☐ failed ☐ by you

Notre job semble s'être correctement exécuté,

Synchronisation Baie

Action in 15h55m

Synchronisation des informations concernant les baies entre racktables et itop

Summary

Report

Log Output

Definition

Node Summary

COMPLETE	FAILED	INCOMPLETE	NOT STARTED
100% 1/1	0	0	0

Node

racktables-server

All Steps OK

script baie

OK

1:34:30 pm

0.00:01

13:34:31 baie_rack.csv DELETED

13:34:31 baie.csv DELETED

13:34:31 itop_server.csv DELETED

13:34:31 itop_rack.csv DELETED

13:34:31 <subprocess.Popen object at 0x7fcc70707fd0> server name, id and rack exported successfully from racktables

13:34:31 <subprocess.Popen object at 0x7fcc70707d90> rack name and id exported successfully from racktables

13:34:31 <subprocess.Popen object at 0x7fcc70718050> server name and id exported successfully from itop

13:34:31 <subprocess.Popen object at 0x7fcc70718210> rack name and id exported successfully from itop

13:34:31 Baie info of server smtpout1 was added to the database

13:34:31 Baie info of server vega was added to the database

13:34:31 Baie info of server ldap15 was added to the database

13:34:31 Baie info of server app2-ts was added to the database

13:34:31 Baie info of server appl-ts was added to the database

13:34:31 Baie info of server ldap-ts was added to the database

13:34:31 Baie info of server mailproxya was added to the database

13:34:31 Baie info of server ldap13 was added to the database

13:34:31 Baie info of server ldap12 was added to the database

13:34:31 Baie info of server ldap11 was added to the database

13:34:31 Baie info of server ldape1 was added to the database

13:34:31 Baie info of server ldape2 was added to the database

13:34:31 Baie info of server hv2 was added to the database

Maintenant que vous savez installer Rundeck et créer un job, vous pourrez remplacer peu à peu les centaines de tâches cron qui moisissent sur vos serveurs. RunDeck est génial car il permet de créer plusieurs projets en fonction de ses besoins (Dev, Ops, Tests ...) et de gérer plein de nodes ce qui ajoute une grosse couche d'automatisation aux tâches récurrentes. Il est également possible d'installer plein de plugins qui étendent les possibilités de RunDeck (Saltstack, Puppet, Hipchat, Chef, Fluentd ...).

lien utile: <https://blog.ouvrard.it/2016/02/25/gestion-de-taches-centralisees-avec-rundeck/>