

Installation d'Ansible

Version 1.0 du 18 Décembre 2016

Référence document : Installation Ansible.doc

	Installation d'Ansible, outil de déploiement automatisé	
--	--	--

Evolutions successives

Evolution	Date	Objet de l'évolution	Référence fichier	Rédacteur(s)
1.0	18/12/2016	Rédaction du document	<i>Installation Ansible.doc</i>	Eric LEGBA

		Date : 18/12/2016	Page : 2 / 21
--	--	----------------------	------------------

	Installation d'Ansible, outil de déploiement automatisé	
--	--	--

SOMMAIRE

1	Préambule	4
2	Introduction.....	5
2.1	Objet du document.....	5
3	Présentation d'Ansible	6
3.1	Problématique d'installation des serveurs et de déploiement des applications	6
3.2	Ansible – Définition et présentation	6
3.3	Environnement.....	7
4	Installation et configuration	9
4.1	Pré-requis	9
4.2	Installation.....	9
4.2.1	Installation à partir du système de paquet Debian	9
4.2.2	Installation à partir du repository d'Ansible sur Github	10
4.3	Configuration des comptes utilisateurs de déploiement.....	13
4.3.1	Création du compte utilisateur «deploy» sur la machine serveur Ansible	13
4.3.2	Création du compte utilisateur «deploy» sur chaque serveur distant	15
4.3.3	Copie de la clé publique de l'utilisateur «deploy» sur les serveurs distants	15
5	Projet Ansible de test	16
5.1	Création d'un projet Ansible	16

		Date :	Page :
		18/12/2016	3 / 21

	Installation d'Ansible, outil de déploiement automatisé	
--	--	--

1 Préambule

Avant de courir, il faut d'abord apprendre à marcher. En vertu de ce principe, avant d'utiliser Ansible, il est conseillé de connaître les notions basiques d'administration d'un système Unix/Linux. Ainsi, vous comprendrez mieux ce qu'est Ansible et son utilité. Ceci n'est qu'un avis personnel et vous n'êtes pas obligé d'y souscrire.

		Date : 18/12/2016	Page : 4 / 21
--	--	----------------------	------------------

	Installation d'Ansible, outil de déploiement automatisé	
--	--	--

2 Introduction

2.1 Objet du document

Dans ce document, nous allons **présenter l'installation d'Ansible, outil de déploiement automatisé de serveurs.**

		Date : 18/12/2016	Page : 5 / 21
--	--	----------------------	------------------

	Installation d'Ansible, outil de déploiement automatisé	
--	--	--

3 Présentation d'Ansible

3.1 Problématique d'installation des serveurs et de déploiement des applications

Pour fournir des services aux utilisateurs, les entreprises déploient des serveurs informatiques qui hébergent des applications. L'installation et la configuration des serveurs et des applications sont des tâches consubstantielles aux activités des intégrateurs d'applications.

Pour les équipes d'intégration, il est pénible de répéter les opérations d'installation et de configuration de façon manuelle. Pour y remédier, ils ont souvent recours à des scripts d'installation pour automatiser les tâches récurrentes. Cependant, il est difficile de maintenir et de faire évoluer les scripts d'installation. Les fichiers de configuration sont parfois nombreux. Aussi, la procédure d'installation d'une application sur un serveur de type HP-UX peut être différent sur un serveur de type Debian.

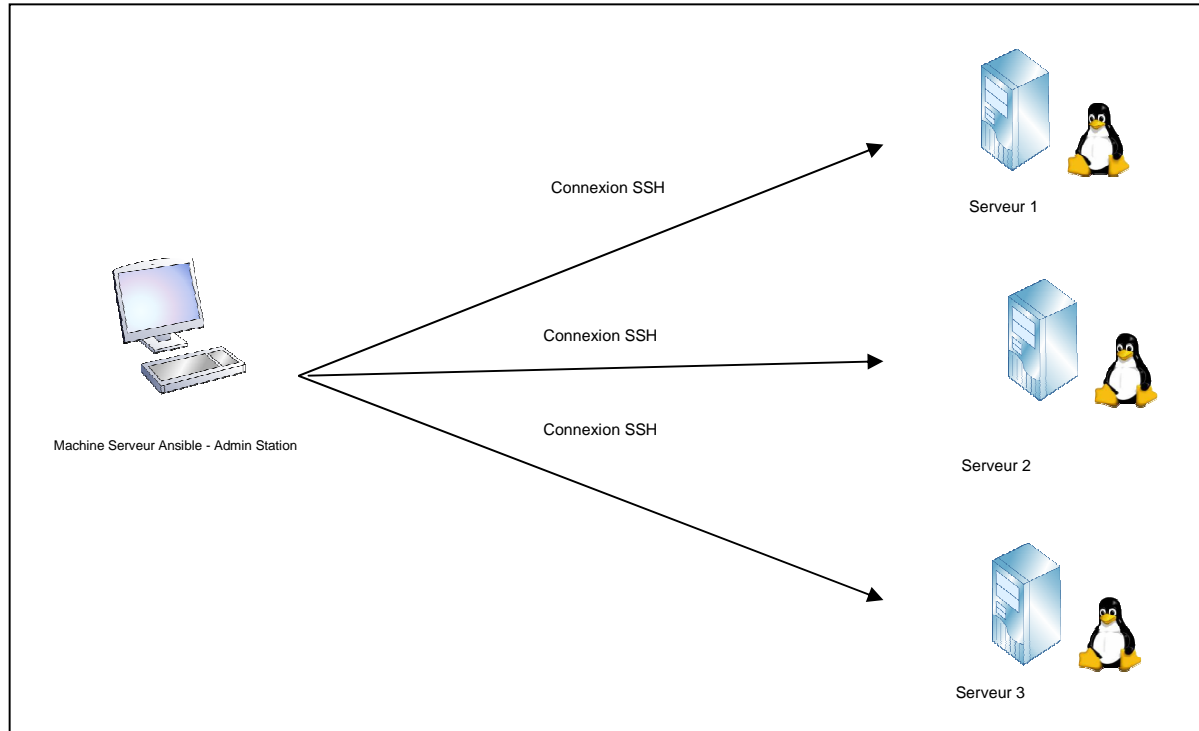
Au vue de ces complexités et des difficultés de maintenance et d'évolution, les scripts d'installation présentent leurs limites. Pour répondre à ces besoins, nous pouvons utiliser Ansible.

3.2 Ansible – Définition et présentation

Ansible est un logiciel open-source écrit en Python dédié au déploiement et à la configuration de serveurs à distance. Il permet d'automatiser les tâches récurrentes d'administration système. Ces tâches répétitives sont écrites dans des fichiers YAML appelés « Playbooks ». C'est le contenu de ces « Playbooks » qu'Ansible exécute sur les serveurs distants. L'une des force d'Ansible est sa capacité à déployer plusieurs serveurs (groupe d'hôtes) simultanément.

		Date : 18/12/2016	Page : 6 / 21
--	--	----------------------	------------------

Installation d'Ansible, outil de déploiement automatisé



Ansible n'utilise pas de clients, ni d'agents sur les serveurs hôtes distants. Il nécessite une machine serveur disposant d'un système Linux. Depuis sa machine serveur (la tour de contrôle), Ansible utilise le protocole SSH pour se connecter aux serveurs distants et exécute le contenu des « Playbooks ».

3.3 Environnement

Pour cette installation, Ansible sera installé sur un serveur Debian 8.3 dont les caractéristiques sont :

```
root@admin-station:~# uname -a
Linux st-debian 3.16.0-4-amd64 #1 SMP Debian 3.16.7-ckt20-1+deb8u3 (2016-01-17) x86_64
GNU/Linux
```

`uname -a` (*-a comme all*) : cette commande donne les informations sur la version du noyau Linux et l'architecture (32 ou 64 bits) du système.

```
root@admin-station:~# lsb_release -a
Distributor ID: Debian
Description:    Debian GNU/Linux 8.3 (jessie)
Release:       8.3
```

		Date :	Page :
		18/12/2016	7 / 21

	Installation d'Ansible, outil de déploiement automatisé	
--	--	--

Codename: jessie

`lsb_release -a` : cette commande permet de connaître la version de Linux installée sur la machine.

		Date : 18/12/2016	Page : 8 / 21
--	--	----------------------	------------------

	Installation d'Ansible, outil de déploiement automatisé	
--	--	--

4 Installation et configuration

4.1 Pré-requis

Avant d'installer Ansible, assurez-vous que «python» et que l'éditeur «vim» soient installés sur la machine. Sinon, installer ceux-ci avec la commande :

```
root@admin-station:~# apt install python vim
```

Ansible étant écrit dans le langage Python, il est primordial de l'installer sur la machine. Ensuite, vim sera utile plus tard pour créer des mot de passe avec Ansible-Vault.

4.2 Installation

Il existe plusieurs façons d'installer Ansible.

4.2.1 Installation à partir du système de paquet Debian

Pour installer Ansible sous Debian, taper la commande ci-dessous :

```
root@admin-station:~# apt install ansible
```

L'installation se fait tout seul. Pour permettre à Ansible d'utiliser la commande «scp» pour les transferts de fichiers distants, il faut dé-commenter la ligne suivante dans le fichier «/etc/ansible/ansible.cfg».

```
scp_if_ssh = True // Décommenter cette ligne en supprimant # qui se trouve en début de ligne
```

Cette méthode d'installation installe le dernier paquet stable d'Ansible sous le Debian. Cependant, cette version n'est pas nécessairement la dernière version d'Ansible. Depuis la création de ce paquet Debian, il y a de fortes chances qu'Ansible ait été déjà mis à jour.

		Date :	Page :
		18/12/2016	9 / 21

	<h2>Installation d'Ansible, outil de déploiement automatisé</h2>	
--	--	--

4.2.2 Installation à partir du repository d'Ansible sur Github

Pour obtenir la dernière version d'Ansible, nous allons installer Ansible en récupérant son code source disponible sur [Github](https://github.com/ansible/ansible).

- Installation des prérequis et des dépendances : pour cette méthode d'installation, en plus des prérequis, il est de notre devoir d'installer les dépendances nécessaires au fonctionnement d'Ansible. Comme prérequis, il faut installer «python» et l'éditeur «vim». Les dépendances sont «python-pip, python-dev, git»

```
root@admin-station:~# apt update
root@admin-station:~# apt install python vim
root@admin-station:~# apt install python-pip python-dev git -y
```

- Installation des librairies «PyYAML, jinja2, paramiko».

```
root@admin-station:~# pip install PyYAML jinja2 paramiko
```

- Créer le répertoire «/opt» s'il n'existe pas.

```
root@admin-station:~# mkdir /opt
```

- Se placer dans le répertoire, récupérer le code source d'Ansible sur Github, exécuter les commandes pour l'installation d'Ansible

```
root@admin-station:~# cd /opt
root@admin-station:~# git clone https://github.com/ansible/ansible.git --recursive
root@admin-station:~# cd ansible
root@admin-station:~# git submodule update --init --recursive
root@admin-station:~# make
root@admin-station:~# make install
root@admin-station:~# make clean
root@admin-station:~# mkdir /etc/ansible
root@admin-station:~# cp examples/hosts /etc/ansible
```

Le fichier «/etc/ansible/ansible.cfg» n'est pas créé automatiquement. Vous pouvez l'initialiser avec cet exemple :

Installation d'Ansible, outil de déploiement automatisé

```
# config file for ansible -- http://ansible.com/
# =====

# nearly all parameters can be overridden in ansible-playbook
# or with command line flags.  ansible will read ANSIBLE_CONFIG,
# ansible.cfg in the current working directory, .ansible.cfg in
# the home directory or /etc/ansible/ansible.cfg, whichever it
# finds first

[defaults]

# some basic default values...

hostfile      = /etc/ansible/hosts
library       = /usr/share/ansible
remote_tmp    = $HOME/.ansible/tmp
pattern       = *
forks         = 5
poll_interval = 15
#sudo_user    = root
#ask_sudo_pass = True
#ask_pass     = True
transport     = smart
remote_port   = 22
module_lang   = C

# plays will gather facts by default, which contain information about
# the remote system.
#
# smart - gather by default, but don't regather if already gathered
# implicit - gather by default, turn off with gather_facts: False
# explicit - do not gather by default, must say gather_facts: True
gathering     = implicit

# additional paths to search for roles in, colon separated
#roles_path   = /etc/ansible/roles

# uncomment this to disable SSH key host checking
#host_key_checking = False

# change this for alternative sudo implementations
#sudo_exe     = sudo

# what flags to pass to sudo
#sudo_flags   = -H

# SSH timeout
timeout       = 10

# default user to use for playbooks if user is not specified
# (/usr/bin/ansible will use current user as default)
#remote_user  = root

# logging is off by default unless this path is defined
# if so defined, consider logrotate
#log_path     = /var/log/ansible.log

# default module name for /usr/bin/ansible
#module_name  = command

# use this shell for commands executed under sudo
# you may need to change this to bin/bash in rare instances
# if sudo is constrained
#executable   = /bin/sh

# if inventory variables overlap, does the higher precedence one win
# or are hash values merged together?  The default is 'replace' but
# this can also be set to 'merge'.
#hash_behaviour = replace

# list any Jinja2 extensions to enable here:
#jinja2_extensions = jinja2.ext.do, jinja2.ext.i18n

# if set, always use this private key file for authentication, same as
# if passing --private-key to ansible or ansible-playbook
#private_key_file = /path/to/file

# format of string {{ ansible_managed }} available within Jinja2
# templates indicates to users editing templates files will be replaced.
# replacing {file}, {host} and {uid} and strftime codes with proper values.
ansible_managed = Ansible managed: {file} modified on %Y-%m-%d %H:%M:%S by {uid} on {host}

# by default, ansible-playbook will display "Skipping [host]" if it determines a task
# should not be run on a host.  Set this to "False" if you don't want to see these "Skipping"
# messages.  NOTE: the task header will still be shown regardless of whether or not the
# task is skipped.
#display_skipped_hosts = True

# by default (as of 1.3), Ansible will raise errors when attempting to dereference
# Jinja2 variables that are not set in templates or action lines.  Uncomment this line
```

Date :

18/12/2016

Page :

11 / 21

Installation d'Ansible, outil de déploiement automatisé

```
# to revert the behavior to pre-1.3.
#error_on_undefined_vars = False

# by default (as of 1.6), Ansible may display warnings based on the configuration of the
# system running ansible itself. This may include warnings about 3rd party packages or
# other conditions that should be resolved if possible.
# to disable these warnings, set the following value to False:
#system_warnings = True

# by default (as of 1.4), Ansible may display deprecation warnings for language
# features that should no longer be used and will be removed in future versions.
# to disable these warnings, set the following value to False:
#deprecation_warnings = True

# set plugin path directories here, separate with colons
action_plugins      = /usr/share/ansible_plugins/action_plugins
callback_plugins    = /usr/share/ansible_plugins/callback_plugins
connection_plugins  = /usr/share/ansible_plugins/connection_plugins
lookup_plugins      = /usr/share/ansible_plugins/lookup_plugins
vars_plugins        = /usr/share/ansible_plugins/vars_plugins
filter_plugins      = /usr/share/ansible_plugins/filter_plugins

# don't like cows?  that's unfortunate.
# set to 1 if you don't want cowsay support or export ANSIBLE_NOCOWS=1
#nocows = 1

# don't like colors either?
# set to 1 if you don't want colors, or export ANSIBLE_NOCOLOR=1
#nocolor = 1

# the CA certificate path used for validating SSL certs. This path
# should exist on the controlling node, not the target nodes
# common locations:
# RHEL/CentOS: /etc/pki/tls/certs/ca-bundle.crt
# Fedora      : /etc/pki/ca-trust/extracted/pem/tls-ca-bundle.pem
# Ubuntu      : /usr/share/ca-certificates/cacert.org/cacert.org.crt
#ca_file_path =

# the http user-agent string to use when fetching urls. Some web server
# operators block the default urllib user agent as it is frequently used
# by malicious attacks/scripts, so we set it to something unique to
# avoid issues.
#http_user_agent = ansible-agent

[paramiko_connection]

# uncomment this line to cause the paramiko connection plugin to not record new host
# keys encountered. Increases performance on new host additions. Setting works independently of the
# host key checking setting above.
#record_host_keys=False

# by default, Ansible requests a pseudo-terminal for commands executed under sudo. Uncomment this
# line to disable this behaviour.
#pty=False

[ssh_connection]

# ssh arguments to use
# Leaving off ControlPersist will result in poor performance, so use
# paramiko on older platforms rather than removing it
#ssh_args = -o ControlMaster=auto -o ControlPersist=60s

# The path to use for the ControlPath sockets. This defaults to
# "%(directory)s/ansible-ssh-%%h-%%p-%%r", however on some systems with
# very long hostnames or very long path names (caused by long user names or
# deeply nested home directories) this can exceed the character limit on
# file socket names (108 characters for most platforms). In that case, you
# may wish to shorten the string below.
#
# Example:
# control_path = %(directory)s/%%h-%%r
#control_path = %(directory)s/ansible-ssh-%%h-%%p-%%r

# Enabling pipelining reduces the number of SSH operations required to
# execute a module on the remote server. This can result in a significant
# performance improvement when enabled, however when using "sudo:" you must
# first disable 'requiretty' in /etc/sudoers
#
# By default, this option is disabled to preserve compatibility with
# sudoers configurations that have requiretty (the default on many distros).
#
#pipelining = False

# if True, make ansible use scp if the connection type is ssh
# (default is sftp)
#scp_if_ssh = True

[accelerate]
accelerate_port = 5099
accelerate_timeout = 30
accelerate_connect_timeout = 5.0
```

Date :

18/12/2016

Page :

12 / 21

Installation d'Ansible, outil de déploiement automatisé

```
# The daemon timeout is measured in minutes. This time is measured
# from the last activity to the accelerate daemon.
accelerate_daemon_timeout = 30

# If set to yes, accelerate_multi_key will allow multiple
# private keys to be uploaded to it, though each user must
# have access to the system via SSH to add a new key. The default
# is "no".
#accelerate_multi_key = yes
```

Comme précédemment, pour permettre à Ansible d'utiliser la commande «scp» pour les transferts de fichiers, il faut s'assurer que la ligne suivante est dé-commentée dans le fichier «/etc/ansible/ansible.cfg».

```
scp_if_ssh = True // Décommenter cette ligne en supprimant # qui se trouve en début de
ligne
```

Enfin, nous allons installer «ansible-lint», outil pour contrôler la syntaxe du code Ansible.

```
root@admin-station:~# pip install ansible-lint
```

4.3 Configuration des comptes utilisateurs de déploiement

Ansible doit se connecter aux serveurs distants et effectuer des actions sur ceux-ci via SSH. Pour cela :

- Sur la machine serveur Ansible, nous devons créer un compte utilisateur qui sera dédié aux déploiement via Ansible. Ce compte s'appellera «deploy» sur la machine serveur Ansible.
- Le compte utilisateur «deploy» qu'Ansible utilisera pour se connecter aux serveurs distants doit être créé sur chacun des serveurs distants. En plus, l'utilisateur «deploy» doit avoir les droits «sudo» sur chaque serveur distant.
- La clé publique SSH du compte utilisateur «deploy» doit être ajoutée à la liste des clés autorisées sur chaque serveur distant.

4.3.1 Création du compte utilisateur «deploy» sur la machine serveur Ansible

Sur la machine serveur Ansible (la tour de contrôle), nous allons créer l'utilisateur «deploy».

```
root@admin-station:~# adduser deploy
```

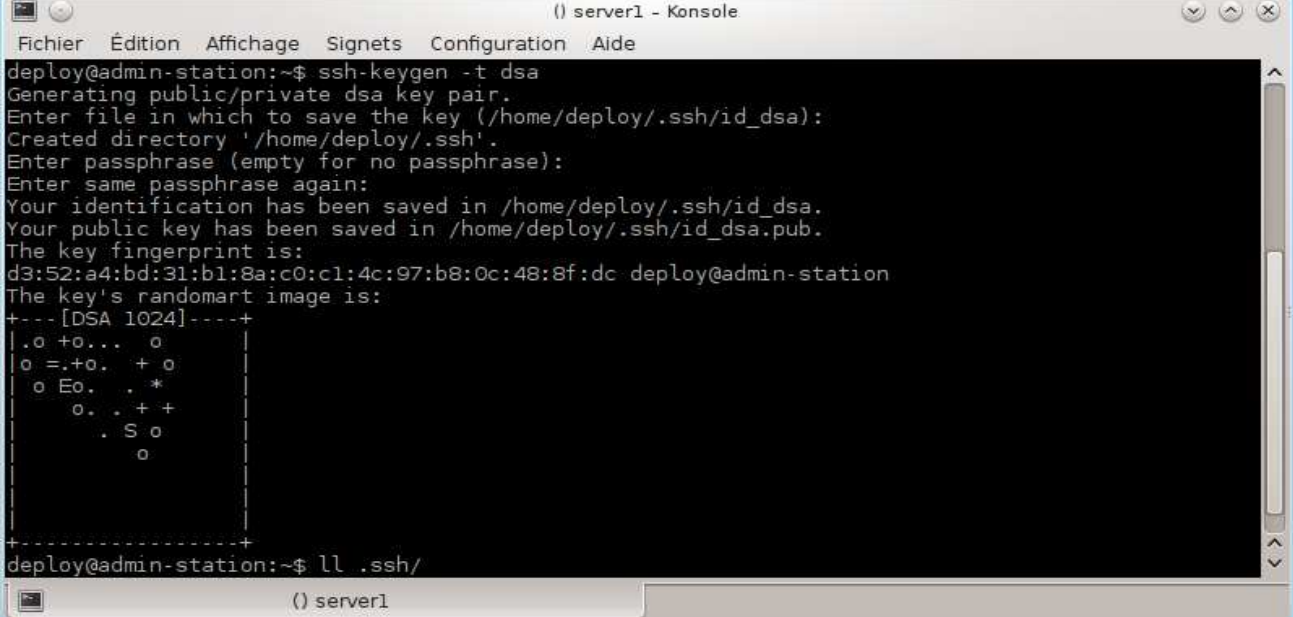
Il faut générer une paire de clés SSH pour cet utilisateur. Pour cela, connecté en tant qu'utilisateur «deploy», exécuter les commandes ci-dessous :

```
root@admin-station:~# su deploy
deploy@admin-station:/root$ cd ~
```

		Date :	Page :
		18/12/2016	13 / 21

Installation d'Ansible, outil de déploiement automatisé

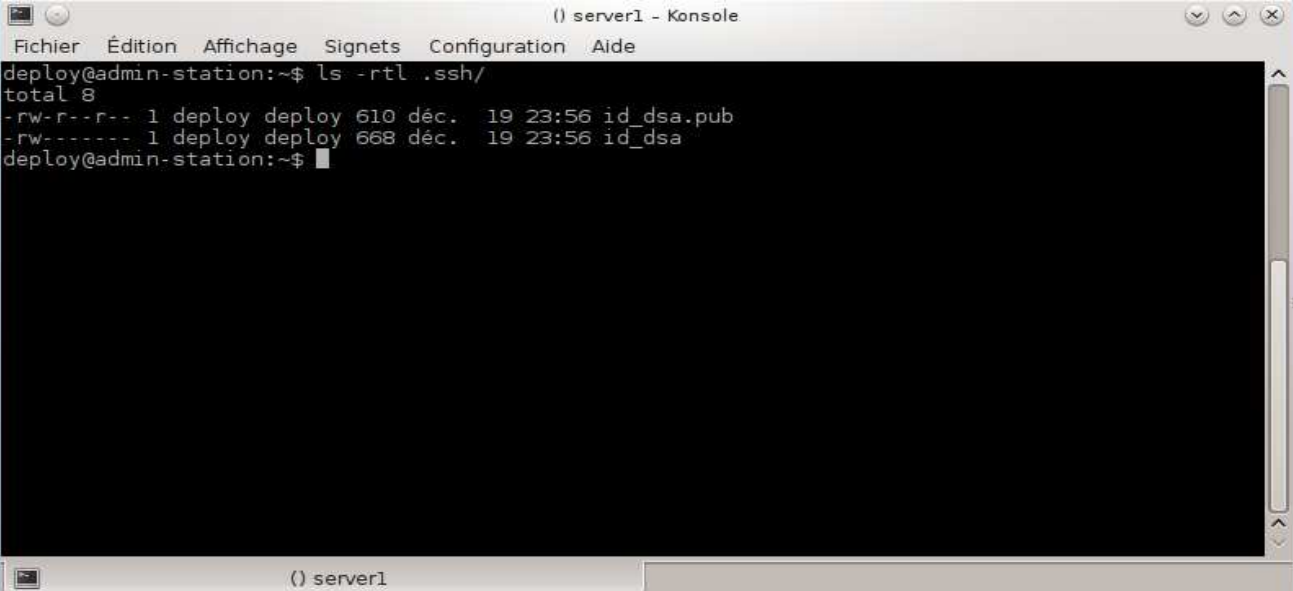
```
deploy@admin-station:~$ ssh-keygen -t dsa
```



```
() server1 - Konsole
Fichier  Édition  Affichage  Signets  Configuration  Aide
deploy@admin-station:~$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/deploy/.ssh/id_dsa):
Created directory '/home/deploy/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/deploy/.ssh/id_dsa.
Your public key has been saved in /home/deploy/.ssh/id_dsa.pub.
The key fingerprint is:
d3:52:a4:bd:31:b1:8a:c0:c1:4c:97:b8:0c:48:8f:dc deploy@admin-station
The key's randomart image is:
+---[DSA 1024]---+
|.o +o...  o
|o =.+o.  + o
|o Eo.    . *
|  o.    . + +
|   .    . S o
|    o
+-----+
deploy@admin-station:~$ ll .ssh/
```

Pendant la création de la paire de clés publique/privée, il vous sera demander une «passphrase» pour sécuriser la clé privée. Pour des raisons de simplicité, on peut ne pas saisir cette «passphrase». Pour vérifier si les clés ont été créées avec succès, on peut regarder le contenu du répertoire «~/ .ssh»

```
deploy@admin-station:~$ ls -lrtl ~/.ssh
```



```
() server1 - Konsole
Fichier  Édition  Affichage  Signets  Configuration  Aide
deploy@admin-station:~$ ls -lrtl .ssh/
total 8
-rw-r--r-- 1 deploy deploy 610 déc. 19 23:56 id_dsa.pub
-rw----- 1 deploy deploy 668 déc. 19 23:56 id_dsa
deploy@admin-station:~$
```

Installation d'Ansible, outil de déploiement automatisé

4.3.2 Création du compte utilisateur «deploy» sur chaque serveur distant

Sur chaque serveur distant sur lequel Ansible se connectera, il faut créer le compte utilisateur «deploy» et lui attribuer les droits «sudo».

Pour chaque serveur distant, en tant qu'utilisateur «root» :

- ajouter l'utilisateur «deploy» en exécutant la commande ci-dessous :

```
root@server1:~# adduser deploy
```

- Attribuer les droits «sudo» à l'utilisateur «deploy»

```
root@server1:~# visudo
```

Ajouter la ligne ci-dessous au fichier ouvert avec «visudo»

```
deploy          ALL=(ALL)        NOPASSWD:  ALL
```

- Initialiser le répertoire «/home/deploy/.ssh» et le fichier «/home/deploy/.ssh/authorized_keys»

```
root@server1:~# su deploy
deploy@server1:/root$ cd
deploy@server1:~$ mkdir ~/.ssh
deploy@server1:~$ chmod -r 700 ~/.ssh
deploy@server1:~$ touch ~/.ssh/authorized_keys
deploy@server1:~$ chmod 600 ~/.ssh/authorized_keys
```

4.3.3 Copie de la clé publique de l'utilisateur «deploy» sur les serveurs distants

Précédemment, nous avons initialisé la clé publique et la clé privée de l'utilisateur «deploy» sur la machine serveur Ansible. Maintenant, nous allons copier la clé publique sur chaque serveur distant.

Connecté sur la machine serveur Ansible (la tour de contrôle) en tant qu'utilisateur «deploy», exécuter la commande ci-dessous :

```
deploy@admin-station:~$ cd ~
deploy@admin-station:~$ ssh-copy-id -i ~/.ssh/id_dsa.pub deploy@server1
```

«ssh-copy-id» va copier la clé publique «~/.ssh/id_dsa.pub» sur le compte utilisateur «deploy» sur le serveur «server».

La configuration d'Ansible et des serveurs distants est terminée. Nous allons tester l'installation en créant un projet Ansible de test.

		Date :	Page :
		18/12/2016	15 / 21

	<h1>Installation d'Ansible, outil de déploiement automatisé</h1>	
--	--	--

5 Projet Ansible de test

5.1 Création d'un projet Ansible

Dans cet exemple, nous allons créer un projet Ansible «hello_world_ansible». Il s'agit d'un simple programme Ansible qui initialisera un fichier «hello_world_from_ansible.txt» contenant la date et l'heure courante.

Pour cela, en tant qu'utilisateur «deploy» sur la machine serveur Ansible «admin-station», il faut créer le répertoire «~/hello_world_ansible». Ensuite, il faut y initialiser les fichiers «hosts.yml», «main.yml», «run.sh».

```

deploy@admin-station:~$ cd ~
deploy@admin-station:~$ mkdir hello_world_ansible
deploy@admin-station:~$ cd hello_world_ansible
deploy@admin-station:~$ touch hosts.yml
deploy@admin-station:~$ touch main.yml
deploy@admin-station:~$ touch run.sh
deploy@admin-station:~$ chmod +x run.sh

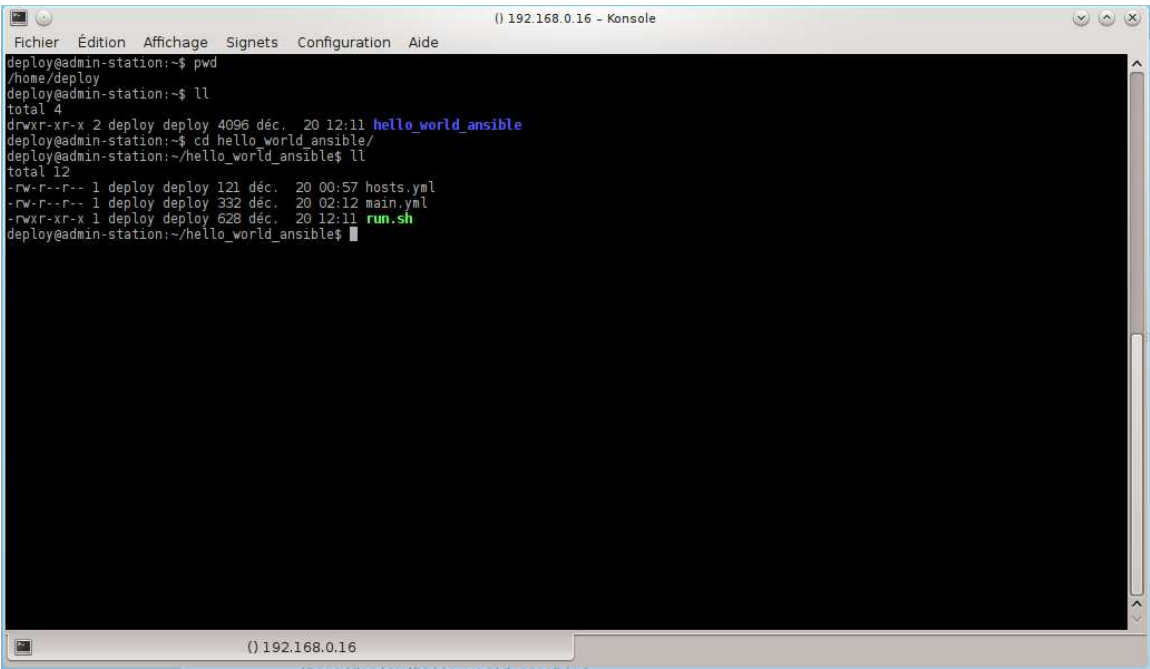
```

À cette étape du projet, le répertoire «~/hello_world_ansible» devrait contenir ceci :

```

/home/deploy/hello_world_ansible/
|-- hosts.yml
|-- main.yml
|-- run.sh

```



Installation d'Ansible, outil de déploiement automatisé

- le fichier «hosts.yml» : nous allons y noter la liste des serveurs sur lesquels Ansible devra effectuer des actions. Comme il s'agit d'un exemple, nous allons utiliser un seul serveur. Voici le contenu du fichier «hosts.yml»

```
server1 ansible_host=server1 ansible_connection=ssh ansible_become_user=deploy ansible_become_pass=deploy1
```

- le fichier «main.yml» : il contient le programme principal d'Ansible. Il contient la liste des actions qu'Ansible aura à exécuter sur les serveurs distants. Dans notre cas, nous allons initialiser un fichier «hello_world_from_ansible.txt» qui contient la date et l'heure du jour. Voici son contenu :

```
---
- hosts: server1
  become: yes
  vars :
    file_destination: "/home/deploy/hello_world_from_ansible.txt"
  tasks:
    - name: Create hello world file
      file:
        path: "{{ file_destination }}"
        owner: deploy
        group: deploy
        mode: 0744
        state: touch

    - name: write message into the hello world file
      shell: echo "Hello World !! Current time is $(date)" > {{ file_destination }}
```

- le fichier «run.sh» : il s'agit d'un script Bash qui :
 - initialise un agent SSH
 - exécute le fichier «main.yml» sur le(s) serveur(s) distant(s) indiqué(s) dans «hosts.yml».

```
#!/bin/bash

ssh_id_dsa_key_file=~/.ssh/id_dsa

# Find an eventual ssh-agent and kill them
if [ "$SSH_AGENT_PID" != "" ];then
    eval `ssh-agent -k`
fi

if [ "$SSH2_AGENT_PID" != "" ];then
    kill $SSH2_AGENT_PID
fi

# Launch ssh agent
eval $(ssh-agent)
```

¹ Normalement, «ansible_become_user» et «ansible_become_pass» doivent être déclarés dans un fichier de mot de passe protégé par ansible-vault.

Installation d'Ansible, outil de déploiement automatisé

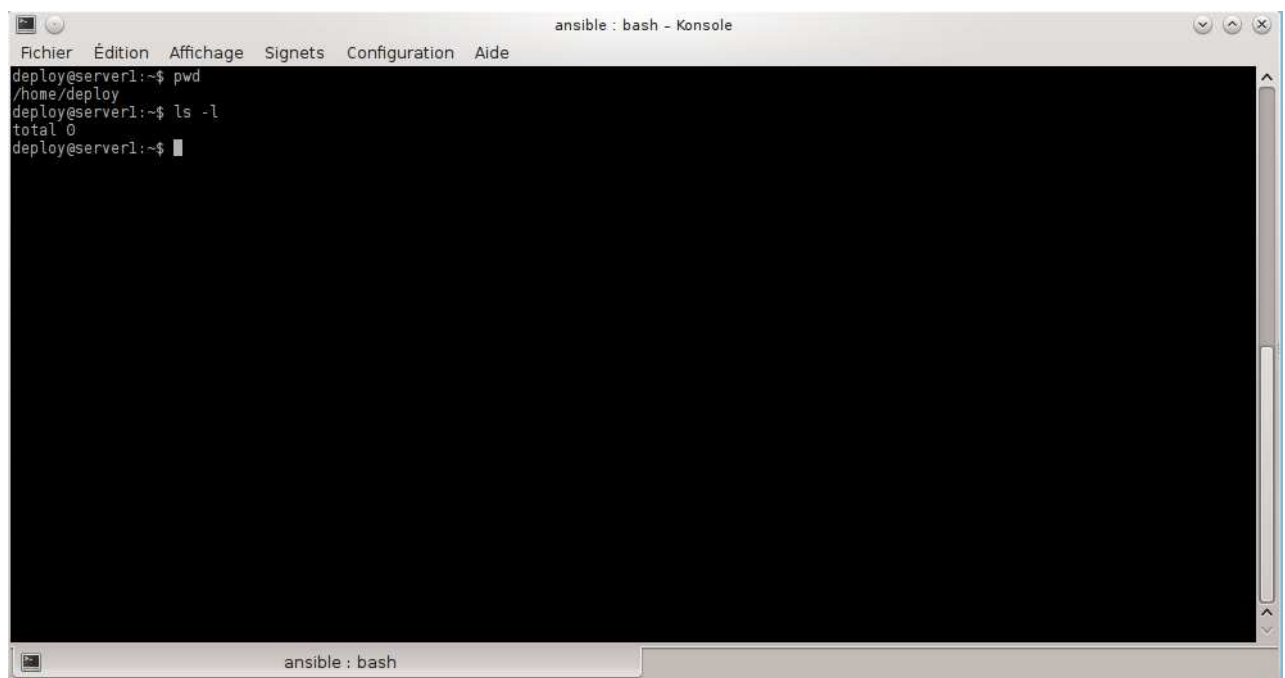
```
echo "-- Ansible version --"
ansible --version
echo "-----"

ansible-playbook -i hosts.yml main.yml -vvvv
```

Avant d'exécuter le programme, nous allons vérifier l'état du serveur «server1» et être sûr que le répertoire «/home/deploy/» ne contient pas le fichier «hello_world_from_ansible.txt».

En tant qu'utilisateur «deploy», se connecter sur le serveur «server1» et exécuter la commande «ls -l ~». Le répertoire ne devrait contenir aucun fichier.

```
deploy@server1:~$ pwd
/home/deploy
deploy@server1:~$ ls -l
total 0
```



Installation d'Ansible, outil de déploiement automatisé

Sur la machine serveur d'Ansible, il faut se connecter en tant qu'utilisateur «deploy», il faut exécuter le script Bash «run.sh».

```
deploy@admin-station:~$ cd ~/hello_world_ansible
deploy@admin-station:~$ ./run.sh
Agent pid 24772
-- Ansible version --
ansible 2.3.0
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/usr/share/ansible']
-----
Using /etc/ansible/ansible.cfg as config file
PLAY [server1] *****
GATHERING FACTS *****
<server1> ESTABLISH CONNECTION FOR USER: deploy
<server1> REMOTE_MODULE setup
<server1> EXEC ssh -C -tt -vvv -o ControlMaster=auto -o ControlPersist=60s -o
ControlPath="/home/deploy/.ansible/cp/ansible-ssh-%h-%p-%r" -o
KbdInteractiveAuthentication=no -o PreferredAuthentications=gssapi-
with-mic,gssapi-keyex,hostbased,publickey -o PasswordAuthentication=no
-o ConnectTimeout=10 server1 /bin/sh -c 'mkdir -p /tmp/ansible-tmp-
1482235255.14-190608725085284 && chmod a+rx /tmp/ansible-tmp-
1482235255.14-190608725085284 && echo /tmp/ansible-tmp-1482235255.14-
190608725085284'

ok: [server1]
TASK: [Create hello world file] *****
<server1> ESTABLISH CONNECTION FOR USER: deploy
<server1> REMOTE_MODULE file group=deploy state=touch
path=/home/deploy/hello_world_from_ansible.txt owner=deploy
<server1> EXEC ssh -C -tt -vvv -o ControlMaster=auto -o ControlPersist=60s -o
ControlPath="/home/deploy/.ansible/cp/ansible-ssh-%h-%p-%r" -o
KbdInteractiveAuthentication=no -o PreferredAuthentications=gssapi-
with-mic,gssapi-keyex,hostbased,publickey -o PasswordAuthentication=no
-o ConnectTimeout=10 server1 /bin/sh -c 'mkdir -p /tmp/ansible-tmp-
1482235258.1-23512627340672 && chmod a+rx /tmp/ansible-tmp-
1482235258.1-23512627340672 && echo /tmp/ansible-tmp-1482235258.1-
23512627340672'

changed: [server1] => {
  "changed": true,
  "dest": "/home/deploy/hello_world_from_ansible.txt",
  "diff": {
    "after": {
      "path": "/home/deploy/hello_world_from_ansible.txt",
      "state": "touch"
    },
    "before": {
      "path": "/home/deploy/hello_world_from_ansible.txt",
      "state": "file"
    }
  },
  "gid": 1004,
  "group": "deploy",
  "invocation": {
    "module_args": {
      "attributes": null,
      "backup": null,
      "content": null,
      "delimiter": null,
      "diff_peek": null,
      "directory_mode": null,
```

Installation d'Ansible, outil de déploiement automatisé

```
        "follow": false,
        "force": false,
        "group": "deploy",
        "mode": 484,
        "original_basename": null,
        "owner": "deploy",
        "path": "/home/deploy/hello_world_from_ansible.txt",
        "recurse": false,
        "regexp": null,
        "remote_src": null,
        "selevel": null,
        "serole": null,
        "setype": null,
        "seuser": null,
        "src": null,
        "state": "touch",
        "unsafe_writes": null,
        "validate": null
    },
    "module_name": "file"
},
"mode": "0744",
"owner": "deploy",
"size": 60,
"state": "file",
"uid": 1004
}

TASK: [write message into the hello world file] *****
<server1> ESTABLISH CONNECTION FOR USER: deploy
<server1> REMOTE_MODULE command echo "Hello World !! Current time is $(date)" >
/home/deploy/hello_world_from_ansible.txt #USE_SHELL
<server1> EXEC ssh -C -tt -vvv -o ControlMaster=auto -o ControlPersist=60s -o
ControlPath="/home/deploy/.ansible/cp/ansible-ssh-%h-%p-%r" -o
KbdInteractiveAuthentication=no -o PreferredAuthentications=gssapi-
with-mic,gssapi-keyex,hostbased,publickey -o PasswordAuthentication=no
-o ConnectTimeout=10 server1 /bin/sh -c 'mkdir -p /tmp/ansible-tmp-
1482235258.51-136326964127186 && chmod a+rx /tmp/ansible-tmp-
1482235258.51-136326964127186 && echo /tmp/ansible-tmp-1482235258.51-
136326964127186'
changed: [server1] => {
    "changed": true,
    "cmd": "echo \"Hello World !! Current time is $(date)\" >
/home/deploy/hello_world_from_ansible.txt",
    "delta": "0:00:00.006311",
    "end": "2016-12-20 17:11:46.966388",
    "invocation": {
        "module_args": {
            "_raw_params": "echo \"Hello World !! Current time is $(date)\" >
/home/deploy/hello_world_from_ansible.txt",
            "_uses_shell": true,
            "chdir": null,
            "creates": null,
            "executable": null,
            "removes": null,
            "warn": true
        },
        "module_name": "command"
    },
    "rc": 0,
    "start": "2016-12-20 17:11:46.960077",
```

	<h1>Installation d'Ansible, outil de déploiement automatisé</h1>	
--	--	--

```
"stderr": "",
"stdout": "",
"stdout_lines": [],
"warnings": []
}

PLAY RECAP *****
server1                : ok=3    changed=2    unreachable=0    failed=0
```

À la fin de l'exécution, nous allons regarder le contenu du répertoire «/home/deploy» sur le serveur «server1»

```
deploy@server1:~$ pwd
/home/deploy
deploy@server1:~$ ll -rt
total 4
-rwxr--r-- 1 deploy deploy 76 déc. 20 17:11 hello_world_from_ansible.txt
deploy@server1:~$ more hello_world_from_ansible.txt
Hello World !! Current time is mardi 20 décembre 2016, 17:11:46 (UTC+0100)
deploy@server1:~$
```

Ce programme est un cas pour tester la configuration mise en place. Il faudra aller plus loin pour découvrir les capacités d'Ansible et tout ce qu'il peut apporter (les rôles, les modules, les groupes de machines, etc...) dans le cadre d'un processus de déploiement automatisé. Pour cela, on peut utiliser les ressources en ligne telles que :

- la documentation officielle : <http://docs.ansible.com/>
- des exemples de programme Ansible : <https://github.com/ansible/ansible-examples>
- etc...